Kenneth D Knight, Jr. The Application of XML as a Means of Exchanging Discharge Summaries between Hospital Information Systems. A Master's Paper for the M.S. in I.S. degree. April, 2004. 67 pages. Advisor: Gary Marchionini

Achieving interoperability between two or more disparate systems has long been both a strong desire and difficult challenge to information professionals. To make it even more problematic, attributes of an interoperability solution for one situation might not be sufficient for another. However, with the use of XML technologies, interoperability between systems is becoming an attainable goal. In this paper, using the health care system (specifically, discharge summaries) as a backdrop, I explore the issues surrounding an XML-based interoperability solution. The proposed solution creates a connection between a Microsoft Access 2002 database and an Oracle 9i database using XML as the intermediate data format. This paper explores the ramifications of exchanging health data, the current XML application offerings of the two databases in question, and the specific problems that must be addressed when creating an XML-based interoperability solution. The last section explains decision rationales and presents a general framework of steps for reproducing this solution.

Headings:

Information Systems -- Interoperability

Information Systems -- Health Care

Programming Languages -- XML

Database Systems -- Oracle 9i

Database Systems -- Microsoft Access 2002

THE APPLICATION OF XML AS A MEANS OF EXCHANGING DISCHARGE SUMMARIES BETWEEN HOSPITAL INFORMATION SYSTEMS

by
Kenneth D. Knight, Jr.

A Master's paper submitted to the faculty
of the School of Information and Library Science
of the University of North Carolina at Chapel Hill
in partial fulfillment of the requirements
for the degree of Master of Science in
Information Science.

Chapel Hill, North Carolina

April 2004

Approved by:

_____

Gary Marchionini

Contents

Introduction

Interoperability has been a sore spot for Information Professionals as long as corporate, government, and educational entities have been exchanging information. However, advanced methods of communication and storage have made it possible for disparate systems to interoperate. The following paper will look at interoperability on a general level and attempt to create a successful interoperability scenario using the health care system as a back drop. It is the intention of this paper to use this scenario to develop a general framework for making two disparate systems interoperable, describe some of the XML functionality of the two database systems in question, and expose some of the likely problems one will face when trying to connect two systems.

What is interoperability and why is it important?

Interoperability is defined as "the ability of two or more systems to communicate, execute instructions, receive data and logically incorporate it into its existing data structures, and carry out any other form of interaction" (Moen, 2001, p.50). For the purposes of this paper, we will focus on the ability of two systems to logically share a certain information object.

Interoperability is important because there are no units of work that are completely self-contained. Any professional entity, be it business or otherwise, must interact with other entities in order to carry out certain duties. This would be considered cooperative work. Cooperative work is only carried out by the existence of distributed components that are able to interoperate. As computing technology becomes more advanced, computing systems become more heterogeneous (Wileden, 1991, p. 73). Without interoperability, such systems could not communicate with one another, and could only provide access to their individual domains. Thus, cooperative work would become impossible.

The problem of interoperability is two-pronged. Not only do the technical environments need to be able to exchange information, but there must also be an assurance that there is semantic interoperability of the content between the applications and partners (Kulvatunyou, 2000, p.198). In the end, two systems must agree on both a mechanism for transferring information so that it can be used by both systems, and the actual meaning of the information.

Why use the health care industry?

Every corporate entity has its own system for creating and storing information. Even companies that perform similar functions may have very different ways of operating. Therefore, immediate interoperability amongst such companies is difficult to impossible without some type of technological customization.

The Health Care Industry is no different in this respect. In fact, for many care-giving facilities, interoperability with other facilities is not seen as a benefit worth the investment. Many smaller health care providers and clinics still rely on paper-based documentation to record and store patient information. But, as methods of health care delivery change and patients are treated by a team of care-givers as opposed to one primary care-giver, it becomes necessary to create patient records that are shared amongst care-giver teams (Grimson, Grimson, and Hasselbring, 2000, p.48-9).

The lack of automatic interoperability between medical information systems causes gaps in an institution's access to patient information, which may hinder its ability to administer proper care. For instance, if a physician at a small health clinic would like to see the latest discharge summary health record for a new patient, he would either need to request the record from the patient or contact the patient's prior health clinic and request the record. The prior health clinic would in turn have to produce a physical document and send it on, a service for which they would certainly charge. Upon receipt of the discharge summary document, the new health-care provider would need to decipher it and either save it in a physical file or manually enter in the data to his own information system. The described information exchange system can be very time consuming and expensive. In addition, doctors are not in the business of requesting, deciphering, and integrating information. Thus, sharing information using the current system is quite inefficient, but absolutely necessary. A possible solution for this is to

come up with a system that will easily allow health care providers to electronically transmit information between disparate information systems.

The rest of this paper will be dedicated to describing such a system. The first section will review existing literature on the subject. It will include information about interoperability, existing health care information systems, and the current work being done on creating a standard for electronic medical records. The second section will go over the specific problem that the paper aims to solve, which includes a hypothetical situation of an interoperability need. The third and forth sections include a solution description. The last two sections go over implementation problems, decision rationales, an implementation framework, and conclusions.

Literature Review

In order to research the problem presented in this paper, it was necessary to read papers concerning a variety of subjects. The research began in search of the most recent developments regarding electronic health records. Upon discovering many of the latest developments in this field, it was necessary to examine how XML was currently being integrated into electronic health records, as well as the Health Level Seven's (HL7) input into the situation. Lastly, it was necessary to examine papers that discussed interoperability methods and standards as a general subject (i.e., *not* as it pertained to health care).

Much of the literature regarding the state of health care technology as it pertained to the patient dealt with applications for offering the patient access to his health records as well as assisting the patient in monitoring his own health. *The Journey to e-Health*, *The Impact of CyberHealthcare on the Physician-Patient Relationship*, and *What it Will Take to Create New Initiatives in Health Care* each discussed how technology was making it possible for hospitals to provide patients with better tools for taking care of themselves and communicating with hospitals/caregivers when necessary. Many of these papers promoted the use of electronic medical records, but did not expressly detail the architecture of them.

There were a number of papers that discussed how XML could play a role in producing electronic medical records. *A Proposal for Electronic Medical Records in U.S. Primary Care* was a very interesting paper that detailed the use of electronic medical records (EMR's) in other countries, as well as the benefit that they had provided thus far. *Medical Markup Language (MML) for Hospital Information Interchange* was a paper that came out before the HL7 standards body had begun deciding on its Clinical Document Architecture (CDA). It described a solution that Japan is using which marks health records up using a controlled medical vocabulary. The HL7 ended up using a similar architecture for its CDA. The International Digital Enterprise Alliance published a paper entitled *Designing XML Based Medical Applications* which built on this idea of creating electronic medical records using XML. From these papers, it was clear that XML would be a strong technology in the creation of electronic health records and hospital interoperability solutions.

The HL7 produced the definitive guide to current work that was being done in terms of incorporating XML into patient records, as well as the creation of XML standards in the health care industry. Earlier papers, such as the 1999 Draft of the HL7 *Patient Record Architecture* helped to explain the concept of an electronic medical record. The Patient Record Architecture was changed to the Clinical Document Architecture (CDA). The August 2003 HL7 Newsletter helped to explain the core goals of the CDA initiative, as well as detail the roles of the Health Information Management and Systems Society (HIMSS) and the Health Information Portability and Accountability

Act (HIPAA) in the creation of the CDA. The institution of HIPAA put the burden of providing reasonable privacy and accessibility features on health records. Thus, creating secure, accessible electronic health records became a solution. *XML and HIPAA: The Data Wave of the Future* discussed this implication. The HIMSS, along with the Institute of Medicine (IOM) were both contacted in order to consult on the creation of the CDA. The HIMSS published its *HIMSS Standards Insight: An Analysis of Health Information Standards Development Initiatives* as a guide for creating Electronic Health Record (EHR) standards in July of 2003. Later that same month, the IOM published its *Electronic Health Record Functional Model: Letter Report* which critiqued the conceptual form of the EHR and ultimately had some influence on the resulting architecture. Lastly, the HL7 released documentation and a PowerPoint presentation that outlined its Clinical Document Architecture (CDA) proposal, which included its vocabulary and structure. This was the basis for my solution.

In regards to creating the more general features of the interoperability solution, as well as addressing some of the issues that arose, I looked to a set of papers which focused specifically on interoperability and systems integration. *The SI Challenge in Health Care* focused on system integration for hospitals. Other papers, such as *Mapping the Interoperability Landscape for Networked Information Retrieval, Specification-Level Interoperability, Interoperability as a Means of Articulation Work* and *Interoperability for Digital Libraries Worldwide* all offered information and guidelines for creating interoperability scenarios in general. Other papers, such as *Conceptual Modeling of XML Schemas, Communication Design for Electronic Negotiations on the Basis of XML*

Schema, and *An Extensible Knowledge Base Management System for Supporting Rule-based Interoperability among Heterogeneous Systems* each discussed the use of XML and XML Schemas in the creation of interoperability scenarios and system integration. The combination of the information from all of these articles assisted in the conceptualization of a solution that would fit the problem put forward by this paper. Lastly, a paper entitled *Uniform Data Standards for Capturing Patient Medical Record Information at Point of Care* was helpful in terms of giving a more detailed description of the data required by a patient record.

The solution for the proposed interoperability problem required information about electronic health care records, the extensible mark-up language, and the interoperability / system integration solutions in general. The papers mentioned here and in the bibliography each offered vital information which played a role in the creation of the proposed solution.

Problem Scenario

Complete open integration between any pair of medical information systems in regards to all information objects that each one stores would be impossible. However, it is possible to drill down and work on integrating a few specific pieces from each system. The following scenario will explain the problem that the described system proposes to solve.

John Benton has just moved from Dallas, Texas to Greensboro, North Carolina. John possesses no physical documentation of his health records. One day, he experiences pains in his chest, dizziness, and nausea. Though not terribly severe, John believes that he needs to seek medical attention, and schedules an appointment at a local physician. During his check-up with the doctor, John is asked a number of questions about his past condition that he is unable to answer because it had been so long since he last saw a doctor. Dr. Jenkins, John's new doctor at Moses Cone Hospital, asks John to contact Methodist Hospital of Dallas and have them produce John's last discharge summary in order to assist him in diagnosing John's ailment.

If John Benton were to carry out his doctor's orders, that would require him to contact his old hospital and wait on a discharge summary to arrive in the mail. Under the proposed interoperability solution, John's old hospital will be able to create and send an

electronic discharge summary to his new hospital. His new hospital will be able to print out the summary and incorporate it into its existing database. The problem can be broken down into a set of sub-problems, which are associated below with the entity that must solve it:

1. Agree on a standard data object structure that will serve as the intermediate between the two systems (both hospitals)
2. Export patient's medical record as an XML document (Methodist Hospital)
3. Map this document to the standard data object structure (Methodist Hospital)
4. Receive, store, and print out this XML document (Moses Cone Hospital)
5. Map XML data to existing relational database table(s) (Moses Cone Hospital)

Solution Part 1: Exporting

In order for this interoperation to work, the two hospitals need to achieve *type-based compatibility*. That is to say, the two information trading partners (hospitals, in this case) must agree on a standard data object structure and meaning, as well as an intermediate format for tagging and transporting the data. This first section of the solution will discuss the use of standards in order to map two abstract data structures to one another.

Type-based compatibility denotes that when two interoperating components are exchanging like data, they must have "consistent views of whatever properties they mutually rely upon that are associated with objects of that type" (Wileden, 1991, p. 75). In order to carry this out, first, we must describe the architecture of the database object that stores discharge summary information for Methodist Hospital of Dallas. The database entity diagram should describe the discharge summary entity as it is held in the information repository of the Methodist Hospital of Dallas.

Since Methodist Hospital of Dallas is not a large hospital and only serves a moderate number of patients, mostly for out-patient treatment, their records are not terribly extensive. Most records are stored into a database, with most of the data stored

as blobs and existing in one table, identified by a case number, and including only one foreign key (Patient Number). Thus, this is a two-tiered data object if you include the foreign key and the information to which it refers. (Please refer to Appendix A, fig. 1-3)

The Moses Cone Hospital database is a little different structurally. This discharge summary is also two-tiered, but the discharge summary notes are broken up into subjects. The data fields are more specific in this data object.

In order for one database to be able to understand information from another database, a type-based mapping must occur. One of the most frequently used approaches for this, and the one that will be used for this solution, will be based on "explicitly translating shared objects to and from a single universal representation" (Wilden, 1991, p. 75-6).

Creating a Standard

Fortunately, there exists the HL7 (Health Level Seven) which is a standards body for the health care industry. The HL7 is tasked with, amongst other things, creating standards for health care practitioners. This ranges from creating standard billing forms for family practices all the way to creating standards for electronic transmission of information between large health care organizations. Currently, the HL7 is working in conjunction with HIMSS (Health Information and Management Systems Society) and HIPAA (Health Information Portability and Accountability Act, which is governed by the

Department of Health and Human Services) to create a standard XML schema and semantic structure for electronic health care records. This effort has been going on since the late Nineties and is now starting to produce some results, though the working draft of the standard has not been finalized. Initially, it was referred to as the HL7 PRA (Patient Record Architecture). This became the HL7 XML PRA, and is now referred to as the CDA (Clinical Document Architecture). The following description of the CDA is based on existing information as of completion of this paper. Since this standard has not been passed by the HL7, certain aspects of the architecture may change.

One of the terrific advantages of the CDA is its three-leveled approach. This approach addresses health care practices that have XML processing technology and those that do not. The first architecture level (Level One) is the simplest. A CDA XML message that conforms to Level One is only required to include a structured CDA header. This header includes information about the patient, the provider, the event being documented, sending system information, receiving system information, and other administrative data. This information is always provided using discrete XML elements. These elements (at the basic level) include: id, document_type, origination_dttm, service_actor, service_target, provider, and person. There are additional elements that may be used to further describe the data, but these are the main ones that must be provided.

The body of a CDA level one document may include, at its simplest, a citation to an external file that contains the patient / event information. However, it may include a

more complex scheme of elements that are used to structure the XML data into human-readable content. This structure begins with a *section* element that optionally includes *caption, paragraph, content, list,* and *table* elements. It is foreseeable that a simple stylesheet utilizing XSL (eXtensible Stylesheet Language) could produce a human-readable document from the XML content.

Level two and three CDA documents explore machine-readability. Level two is for low-level machine-readability, and level three is for the highly structured document that is composed specifically for machine-readability and processing.

At this point, the concept of the CDA document has been agreed upon. However, as mentioned before, the HL7 is only as far as the CDA Level One DTD in terms of milestones, and this itself has not been finalized. Thus, the semantics / data dictionary for these documents have not been finalized either. Though the HL7 does have its own datatypes and elements within its overall schemas, the structure for specific documents (such as discharge summaries) has not been decided upon.

In lieu of using an actual HL7 standard schema for creating the discharge summary, a sample standard must be created to stand in. Though creation of standards is not the focus of this paper, its consideration should be noted. The standard for this scenario will need to come in the form of a standard XML schema for a Discharge Summary. The problem that this schema will have, as will that of the HL7 version when its Level Three semantic schema is developed, is that it may have difficulty providing

sufficient expressiveness to support accurate and semantic expression. If the schema is too complex and difficult to understand, then implementers cannot use it. If it is too simple, then it will allow for imprecise specifications, and problems with interoperability will thwart the process completely (Kulvatunyou, p.198).

Although the exact specification for the CDA Level One DTD is still not finalized, I am using the recommendation version as the basis for my CDA Level One XML schema (see Appendix A, figure 6). I am sticking to the exact same structure, and merely omitting a few of the elements that are not necessary for this project. In order to construct a CDA Level Three XML schema, I have built on the Level Once schema. The structure has been altered only slightly to incorporate machine-readable elements. This schema will be used expressly for Discharge Summaries, and thus, only elements specific to the Discharge Summary data object will be included. These elements have been taken from OpenHealth.org group, which is a Canadian group associated with HL7. They have created DTD modules for admission and discharge summaries. I chose my elements from existing elements in the OpenHealth's 'admission.discharge.module.dtd' (see Appendix A, figure 7).

The following list of elements was decided on from the list of element modules in the OpenHealth DTD. This list gives a clear picture of the discharge summary, but does not go into too much detail. A short description of each element is included, as well.

ELEMENT	**DESCRIPTION**

Cardiac	Type of cardiac trouble
Consultation	Other doctors who were consulted for this interaction
Diagnosis	Diagnosis of patient's problem
Discharge_notes	Miscellaneous notes from Discharge Summary
Family_history	Family's medical history, if applicable
Follow_up	Scheduled follow up actions
Hospital_course	Courses of action taken by hospital not classified as procedures
Known_allergies	Allergies of patient, both natural and drug
Laboratory	If laboratory tests were taken, these are the results
Medication	If patient is on medication, it is listed here
Prescription	Medical prescription prescribed by doctor
Problem	Initial description of perceived affliction
Procedure	Hospital procedures executed
Respiratory	List of respiratory problems

These elements were added to the Level One XML schema in order to create a Level Three schema, producing documents for machine-reading. For our purposes, the schema introduced here will be referred to as the *Standard Discharge Summary Schema* (see Appendix A, figure 7).

Exporting data as XML using Access 2002

The actual discharge summary object must be extracted from a database and converted to XML. There are a couple of ways that this can occur. Some databases have built in mechanisms for exporting tables as XML. However, these export methods can be difficult to append. Therefore, a user must use the XML that the database application creates. Microsoft Access 2002 has such a built-in mechanism.

MS-Access 2002 will allow users to export tables and queries as XML. The user, however, is not given any options as to how that XML is structured. Access creates a simple, three-tiered XML document. The grandparent node is always titled *dataroot* and refers to a proprietary Microsoft namespace. The parent node is titled after the table or query name (Microsoft will sometimes add a unique identifier to this element), and all children of that node are titled after the field headings. It is not possible to insert data as an attribute of any of these elements, nor is it possible to include data from other Access tables.

The advantage to this is simplicity. Though a complex database would not benefit from this feature, smaller databases with simple, linear data structures do benefit. However, the disadvantage is that a user will more than likely have to manipulate the exported XML for further processing. Official W3C namespaces will have to be added. The *dataroot* element will likely need to change. Some elements may need to be changed

to attributes. Since none of these actions are built into MS-Access, they must be executed after the XML export.

Another disadvantage to the XML export function is that it only allows the user to export one table at a time. As mentioned before, related tables cannot be exported together as the same XML document. However, MS-Access does offer the ability to export queries. And, since a user can use SQL language to execute queries, it is possible to combine data from multiple tables into one view and export that as an XML document. However, this query-based XML may still need to be manipulated.

A last caveat of using the MS-Access export utility for use with the Standard Discharge Summary Schema involves the *clinical header* section. In order to properly insert data into these required elements, the MS-Access database must utilize a specific, dedicated table that will export Standard Discharge Summary Schema specific data for inclusion in the Discharge Summary XML document. This information cannot exist in the XSLT stylesheet because it is not static. Some of it is dependent upon the receiving entity. In addition to forcing the XML producer to add a dedicated table specifically for XML production, it also denotes that there must be some level of communication and coordination between the two information exchanging partners in order for the exchange to be successful.

In terms of the health care scenario, John Benton's old hospital will need to merge John's personal information with his latest discharge summary and XML schema specific

information, map the results to the Standard Discharge Summary Schema, and send it to John's new hospital. Appendix A, figure 9a shows the SQL statement that was executed to combine data from existing tables into one Discharge Summary query table. Figure 9b shows how this new query table was mapped to XML.

Once the raw Discharge Summary XML document has been produced by the database, it must still be manipulated to match the structure and syntax of the standard Discharge Summary Schema. The easiest way to accomplish this is by using Extensible Stylesheet Language for Transformation (XSLT). XSLT is a technology that allows users to manipulate XML documents using a rule-based stylesheet. The final product can be of multiple file types (i.e. – HTML, XHMTL, PDF, TXT, etc.). For this problem, the final product needs to be XML. XSLT stylesheets can be applied to XML documents in the runtime in order to produce Discharge Summary Schema-compliant XML documents. The only difficulty is creating the stylesheet that can do it. The XSLT stylesheet that converts the raw Discharge Summary XML to a form that is compliant with the Standard Discharge Summary Schema, as well as the conformant Discharge Summary XML output, is shown in Appendix A, figure 10.

Solution Part 2: Importing

Now that the data has changed formats such that it conforms to the agreed upon standard, it must now be incorporated into the Moses Cone Hospital domain, which uses Oracle 9i. This solution includes an explanation of how XML is imported into the Oracle domain, stored natively, and scanned using scripts with XPATH queries. These queries insert the Discharge Summary data into relational tables.

Importing XML into an Oracle 9i database

Oracle 9i offers a great deal of functionality when it comes to XML. Oracle's 9i release comes with a robust set of tools for using XML within its repository. Specifically, Oracle offers an XML Software Development Kit and a component to its database software called XML DB. For the purpose of this system, Oracle offers the following applications:

a) XML parser – validates XML documents against XML schemas
b) FTP / HTTP access – allows XML documents to be FTP'ed to the repository
c) SQL validation – validates XML documents against extra constraints written into the database schema code

d) Schema-level XML to SQL type-mapping – maps XML types to SQL types

e) XML Storage – searchable, native storage of XML documents

f) XPATH querying – uses XPATH expressions to search XML documents and copy element values to relational databases.

There is one issue with Oracle XML to SQL mapping that must be addressed here. Oracle advertises the ability to incorporate XML-to-RDBMS mapping rules within the actual XML Schema using the XDB namespace as tool for architecting this map. In reality, though this process does create complex Oracle objects within the repository, these objects are not mapped directly to a linear RDBMS. Another step must be taken to assemble these objects as a flat data table. In reviewing the options, it made more sense and took less time to natively store the original XML document and use XPATH expressions to extract the necessary data for the RDBMS, rather than using schema-level processing rules to shred incoming XML documents into multiple Oracle objects. The rest of the solution explanation is based on the use of this XPATH strategy.

In order to import XML and map it to a relational table, first the intermediate XML Schema must mapped directly to the database schema. It is important to conceptualize this mapping before implementing it. A user must put the database schema and the XML schema side by side and determine which XML schema elements must go into the database. In regard to our healthcare scenario, the standard Discharge Summary schema does not map element for element to the Moses Cone Hospital Oracle RDBMS. For this

solution strategy, the Moses Cone Hospital DBA's have decided to go with the strategy introduced by Appendix B, Figure 1 and Figure 2.

It was previously believed that the XML data could be taken directly from the schema-conformant Discharge Summary XML document sent by Methodist Hospital with no problem. However, intricacies regarding both the Oracle system as well as the essence of interoperability have shown that this task is easier said than done. In the implementation phase of this solution, it was discovered that the Oracle XDB application required XML documents to include a value for the *xsi:noNamespaceSchemaLocation* and the *xmlns:xsi* attributes that are usually located in the top-level element of an XML document. Oracle uses the *xmlns:xsi* attribute to hard-code in the namespace for the W3C Schema-instance schema. Access 2002 does not hard-code this in, but instead, includes its own proprietary namespace. This hard-coding could be a requirement of the SDSS, but in this case, it is not. It would be impossible to hard-code in the *xsi:noNamespaceSchemaLocation* attribute on the fly. This is a unique location variable set by the Moses Cone system that points to the validating schema. In this case, the location / value of this namespace attribute is:

localhost:8080/home/MOSESCONE/xsd/discharge_summary.xsd

Methodist Hospital has no way of implicitly knowing this. Thus, Moses Cone must manipulate the SDSS-valid XML once it arrives in order to include this attribute. See Figure 3 for further explanation. The answer to this problem was to create a *que* folder. Incoming XML documents rest in the *que*, awaiting some manipulation which

will ready them for use by Oracle. The XML manipulation can be programmatic or manual. Its implementation was out of the scope of this solution.

It was also previously believed that once the XML was imported by Oracle, data element values could be seamlessly and extracted and added to the relational tables. However, one requirement of type-based mapping that is not addressed by the XML schema is *datatype-mapping.* Oracle uses strict datatypes to preserve the integrity of its data. However, datatypes are different from database application to database application. Thus, the discharge summary XML document, though conformant to the SDSS, may not conform to the datatype requirements of the receiving database. For instance, the *date_of_admission* element may allow a date value in the form of MMDDYY. However, the DATE datatype in Oracle requires YYYY-MM-DDTHH:MM:SS. This incompatibility presents another problem. It appears that data must be cleansed before it can reach its final destination. Thus, the data must be extracted from the XML document, then placed in a temporary table that does not incorporate datatypes. From this temporary table, the data must be converted to the necessary datatype, then inserted into the Discharge Summary table, its final destination.

A PL/SQL script can be implemented to execute the XPATH query, data extraction, and data insertion required to get the data into a temporary table, which has been titled: Discharge_Summary_Que. See Appendix 4a and 4b for examples of these scripts. Another script must be executed to check the validity of the data, manipulate the data so it conforms to datatypes, and then insert the data into the final Discharge

Summary table. Once this takes place, the information transfer is complete. Moses Cone has both a native XML document of the discharge summary, as well as a record of the summary in its own relational database.

Problems, Discussion, and Decision Rationales

A number of problems affected both the creation and implementation phase of this system. The following section outlines many of the decisions that were made to address these problems, as well as the rationales behind them.

1. **What datatypes should be used for fields such as Procedure, Diagnosis, Discharge Notes, etc?**

This problem is not addressed in the HL7 Clinical Header DTD. DTD's themselves do not offer very flexible data-typing. Problems came up when XML element values were inserted into the Oracle Discharge Summary table, and the insert failed because the datatype/format was not compatible. To address this issue, the SDSS only uses the *string* datatype. It is up to the entity receiving the data to manipulate it such that it fits a particular datatyping scheme.

In going through this problem, it is my belief that datatypes must be kept very simple in the SDSS. Though the schema I used only incorporates the *string* datatype, the *dateTime* datatype could easily have been used for the *date_of_admission* and *date_of_discharge* elements. Fortunately, in this scenario,

Access 2002 exports ‘dates’ as *timeInstance* datatypes, which are compatible with Oracle date/time datatypes. This ‘fortuitous’ compatibility will be the exception, not the rule in most cases, and thus, should be addressed in the SDSS. But, this will not save a hospital receiving the information the trouble of manipulating it to fit its own repositories; it will just make it easier. To re-iterate, the datatypes need to be kept simple (ex: string, dateTime, integer, etc.) so that each hospital can map their data to/from it without too much difficulty.

2. What is the benefit to using a standard Discharge Summary schema over just connecting directly to the receiving health care system?

More specifically, this question asks why Methodist Hospital doesn’t just build a ‘bridge’ to the Moses Cone Hospital system on order to transfer data, rather than trying to take the extra step of using the SDSS. Creating multiple connections to lots of different systems results in what is referred to as *spaghetti string* interoperability. Each hospital would have to build ‘bridges’ to every other hospital, and be responsible for the information about these ‘bridges’. This could be a hefty bit of work, not to mention costly, especially when the alternative of using the schema standard requires the user to create only one conceptual / technological map.

However, creating a standard is not so easy either. The CDA has been in discussion for over five years, and as of recently, only the *header* section had

actually been created (though, not approved). Regardless of the resources necessary to create a standard, in the healthcare setting, this is the best option. Hospitals should focus on their primary function, which is to be top care-givers, not technological power houses. Creating such standards would both allow the hospital to focus on its primary objective of offering quality health care and facilitate an easy method for electronically receiving health information from other health care providers.

3. **Should the standard schema reflect the structure of the document, or help to describe the contents of the document? Should tags be: <section>, <caption>, <paragraph> OR should they be <discharge_notes>, <cardiac_factors>, <diagnosis>?**

This was an interesting question to ponder when deciding on the solution for this problem. Basically, the question asks if the CDA should be machine-readable or human-readable. It also touches on a larger topic, which questions the ability to offer machine-readability to documents (such as Discharge Summaries) that are more qualitative than quantitative. The answer to this truly depends on the end-users.

Fortunately, the HL7 answered this question by offering three levels to the CDA. Level one, a simpler version, offers only endtags like <section> or <caption>. These documents could incorporate the use of XSL stylesheets to become

comprehendible health documents. With little to no change applied to these documents, a doctor could read one and reasonably get what he needed from it. If this doctor worked in a small hospital that didn't use a complex relational database to store patient records, then this would be enough, and machine-readable tags would not be necessary. However, if this doctor worked in a large hospital that warehoused this type of data, the machine-readable elements may become necessary.

I believe that this problem questions the nature of health care documents. In viewing a number of health care documents and discharge summaries in preparation for this project, the use of notes and paragraphs in these documents was quite evident. The actual information made the most sense when read in the context of the Discharge Summary document (i.e. – the information might make less sense if read from a database interface, separated from other sections of the document). Rarely did any information entity (i.e. – diagnosis, procedure, etc.) consist of a one word value. It was typically a textual string, or paragraph. If this were strictly financial data, it would be much easier to label, break apart, and add to a database. But health records demand some continuity and relation between the separate information entities. This fact makes the addition of a presentation layer for the data a must. It also makes the addition of consistent, logical machine-readability more difficult.

So, to answer this question, I would say that both should be required. I used the CDA Level Three standard, which incorporates both, and I feel it was the right decision.

4. **Complete patient records are normally divided into several linked tables. How does one convert this distributed patient record into an XML document?**

This became an issue when I realized that Access 2002 could not export multiple, linked records as XML documents. The way around this was to create a query that would combine all of the necessary information as one, single record, and then export that record. For this solution, I created a standard XML export query, and only modified the *WHERE* clause in the SQL statement to choose particular patient records.

Another interesting caveat to this interoperability scenario was that there were several fields in the SDSS that did not have counterparts in the Methodist Hospital database. So, in addition to extracting information from the *Patient_Info* table and the *PDS_X* table, a third table was created. Because this data is not static, it did not make sense to change the XSLT stylesheet such that this data was added to the XML export document. This third table, entitled *XML_export_clinical_header_data*, holds typical data that would exist in an SDSS-compliant XML document, but has no place in the regular database schema

of Methodist Hospital. The addition of this table made it much easier to create the schema-compliant Discharge Summary XML document.

5. **There will almost certainly be some discordance between the names of elements that share the same semantic meaning. What if 'allergies' is its own field in one database, but allergy information is simply included in the 'discharge summary notes' field of another database? How should this be addressed?**

This was probably the most difficult conceptual issue to handle when solving this problem. Interoperability between two hospital information systems offers the benefit of information exchange without much human analysis or handling. But, when values for elements with the same name but different semantic meaning are exchanged, or when same element values are stored in different elements, a breakdown in the interoperability occurs. Worst case scenario, integral health information is either dropped or stored in an unlikely spot during the transfer, thus rendering it inaccessible to the health care professional at the receiving hospital.

This issue must be addressed in a number of ways. First of all, no matter how much we would like the interoperability solution to offer a complete hands-off exchange of information, this is just not realistic. In a situation like this, where information can be both qualitative and dependent on its actual presentation, there must be some communication between the hospitals upon initial information

exchange. For example, the CDA Clinical Header DTD includes an element *Service_Target*. This refers to the type of system where the XML document will end up. Moses Cone Hospital would have to manually give this information to Methodist Hospital for this element value to be correct. Also, even though the SDSS does a good job of setting up the required data fields, it does not put a requirement on how the content of these fields is set up. For example, the values for *procedure* could be a LIST, according to one hospital, and a PARAGRAPH, according to another. The communication between the hospitals may be something simple like exchanged emails between database administrators, but it must take place in order to clear up any inconsistencies in the information exchange process.

Secondly, a presentation layer must be added. This should not be a requirement of all interoperability systems, but it should certainly be required of health care interoperability systems. In analyzing this problem, it became clear that health care data was different from other types of data. Its presentation and context adds to its overall meaning. So, there may be some loss of meaning when the Discharge Summary data is inserted into a relational table, then reconstructed in report form. Additionally, not all health care systems have the means of setting up XML extraction methods such that data is automatically cleansed and inserted into a database. But, with the existence of a presentation layer, this 'meaningfulness' could be preserved. Presentation layer, in this scenario, comes

in the form of an XSL stylesheet that styles the XML document for presentation on the web. A sample presentation is included in Appendix B Figure 5.

6. **Should the discharge summary be saved as a BLOB? When a discharge summary is received, shouldn't there be some designation that it was imported?**

It is my belief that discharge summaries should still be saved as BLOB's, marked, and referenced in a hospital's record of the patient interaction. If the discharge summary information, once extracted, is not comprehendible as a result of it being stored in a relational database, it should still be accessible via the original Discharge Summary. If this system were to be implemented, the discharge summary table should include a field that acts as a pointer to the actual location of the XML document.

Regarding the inclusion of some designation that the information was imported, there is an element in the CDA for *originating_organization*. In an actual instance of this interoperability system, this field should be included to designate that the discharge summary information was from a different hospital. This was not implemented in my test system, but should be implemented in a run-time system.

In this system, Oracle does store the entire Discharge Summary XML document natively so that it can be searched and retrieved. It is stored as XMLTYPE, which is a datatype created by Oracle to store XML documents. It can be marked and associated with other relational attributes via this method.

7. Were there any other unexpected problems or issues that arose in the implementation of this system?

There were two other unexpected issues that came up during the creation of this project. The first was the necessity of adding *ques* to hold temporary instances of the XML and Discharge Summary data while it was being altered for schema and/or database compliancy. When Moses Cone received schema-compliant Discharge Summaries in XML form, they had to stay in a que until they could be appended for use in the Moses Cone system. On the Moses Cone side, an additional temporary table had to be created to hold data from the XML document so that it could be 'cleansed' to match the Moses Cone datatypes. This acted as a relational table que. An SQL script must be executed to transform this data to the datatype that is compliant with the existing Discharge Summary table in the Moses Cone system.

The second issue dealt with how Oracle treats XML documents. The extensible markup language is a standard, but there is no standard way of treating it. Oracle had some specific rules for dealing with XML documents, such as requiring

certain namespaces and schema location attributes. These intricacies of dealing with XML are rarely listed, and must be dealt with when setting up a system like this. All databases probably have different ways of dealing XML. But, this made it difficult because the issues usually didn't come up until the implementation phase. For instance, there are cases when an element is EMPTY, but it would have an attribute that would hold a value. There could be an element such as *origination_dttm* that is empty, but possesses a *V* attribute. The *V* attribute would hold the elements actual value. This example occurs in the CDA header DTD. However, Oracle will not validate XML documents that contain empty elements. Therefore, the schema had to be altered slightly to get around this. The *Discharge Summary* schema that is registered and stored in the Oracle system changes this element slightly to account for this. This schema is a near duplicate of the CDA Level Three schema, but as mentioned before, is altered slightly to account for how Oracle deals with XML schemas.

General Solution Framework

Below, I have included a general set of steps for using XML as a possible conduit for creating an interoperability scenario.

PRELIMINARY STEPS

(Executed by all participants in the information exchange)

A. Decide on what data objects are going to be exchanged.
B. Decide on a standard XML structure (schema or DTD) for that data object.
C. Come to some agreement on the semantic meanings of the standard DTD elements and ensure that it will work with existing repository.

STEPS FOR SENDING DATA

1) Study how your database treats XML. What functionality is offered? Can your database insert XML-type features when creating XML documents? It is important to assess these qualities and figure out how they will affect XML production.

2) Write a query or create an application that will assemble all parts of the data object. If your database is able to assemble XML instances from separate tables, then this may not be a necessity. However, the data object must be assembled before creating an XML document.
3) Export the data object as XML, or export data and convert it to XML.
4) Map the flat XML structure to the complex, standard schema structure

STEPS FOR IMPORTING DATA

1) Import the schema-compliant XML data object to a holding que.
2) Manipulate the data such that it fits the 'receiving' database schema
3) Import the natively store XML.
4) Use an XPATH script to extract data from the XML document and insert it into a temporary relational table.
5) Analyze the data in this temporary table, and alter / cleanse it (if necessary).
6) Import this data into the final database table.

Conclusions

This investigation, solution analysis, and implementation exposed a number of issues that will come up in all interoperability scenarios. It became quite clear during the solution analysis phase of this investigation that the solution to this problem would be greatly affected by the information being exchanged. Because medical records and health information include such a great deal of description, they are tough to dissect and label. Thus, labels needed to be created for entire chunks of information, not just simple values. In addition, decisions on how to label and exchange these data elements were affected by the make up of the information itself.

Second, interoperability cannot be successful unless there is some level of coordination and communication between the information exchange partners. Using a schema standard will assist in this interoperability goal. However, the users cannot expect to just map to the standard only, and get the full benefit of interoperability. There will be some gaps that must be filled by actual communication.

Third, it is important to note how each different system treats XML. This will play an important role in the setup of an organization's use of XML. Oracle 9i and

Access 2002 have very different ways of handling XML. These methods must be pre-determined to set up an efficient interoperability scenario.

Fourth, with information such as health data, a presentation layer must be created and included in the exchange. It is possible that implementing an efficient machine-readable solution is just not possible between two systems. However, the inclusion of a standard stylesheet for presenting the XML will help to solve this problem.

Lastly, while the above solution will work for Discharge Summaries within the Oracle 9i and Access 2002, it may not be optimal for other database systems or other types of data. Financial data might work better using a different scenario. However, I believe that this is a good starting place for creating interoperability scenarios for the exchange of health data.

Appendix A

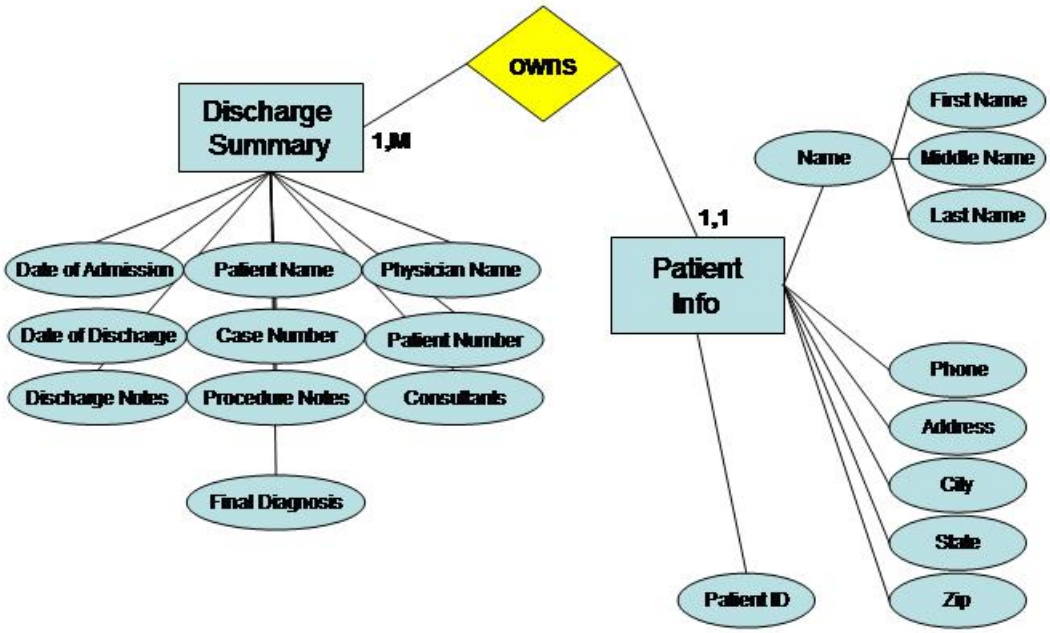
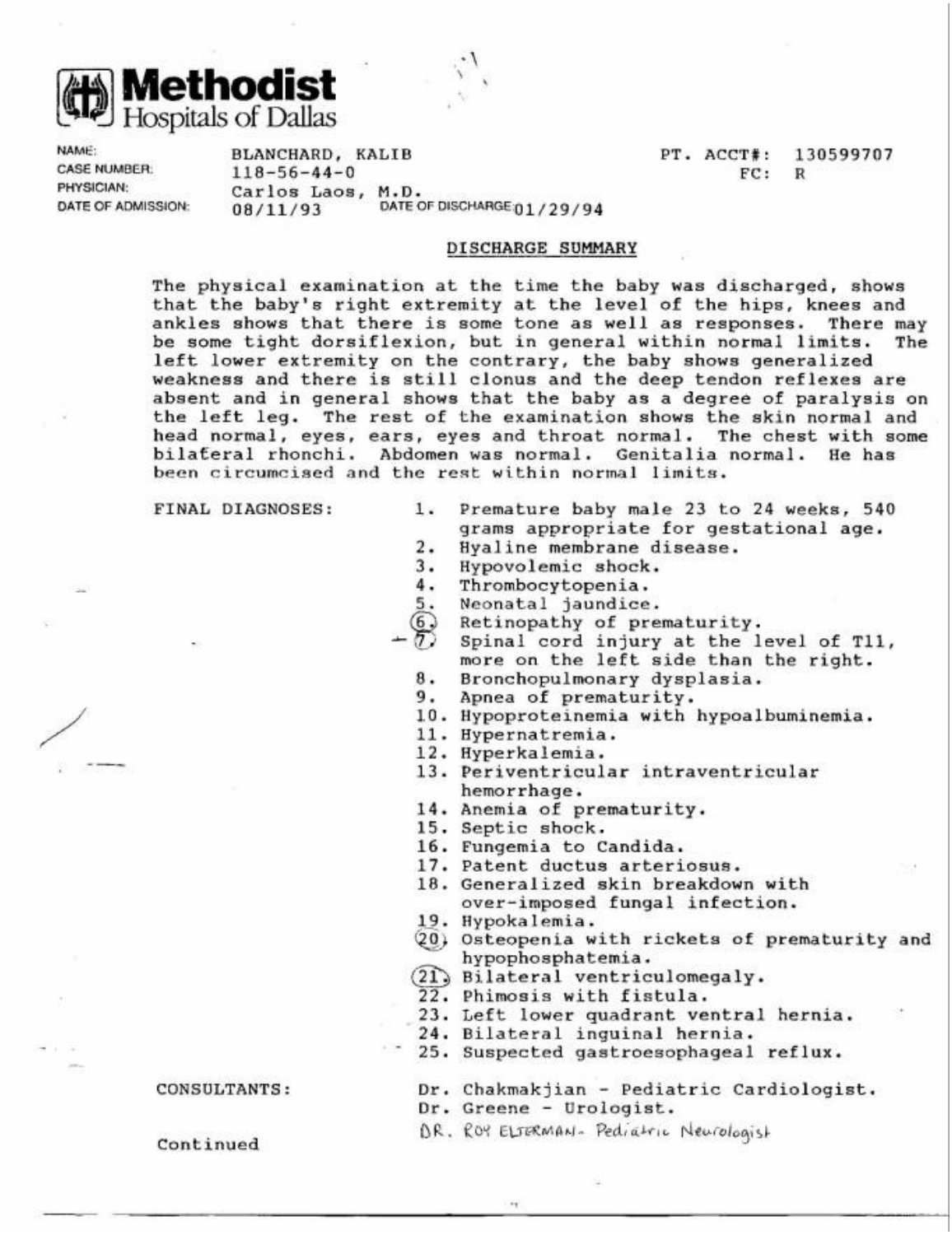


Figure 1: Database Entity Diagram for Methodist Hospital

Figure 1: ER Diagram of Methodist Hospital Data Repository. This figure shows the relationship between the entities and attributes stored in the Methodist Hospital database.

Figures 2a – 2d: Methodist Hospital Discharge Summary. The following figures are scans of an actual discharge summary from Methodist Hospital of Dallas. There were a few pages that included more discharge notes which were not included in these figures. These figures include all information used by the respective data model.



Methodist Hospitals of Dallas

NAME: BLANCHARD, KALIB PT. ACCT#: 130599707
CASE NUMBER: 118-56-44-0 FC: R
PHYSICIAN: Carlos Laos, M.D.
DATE OF ADMISSION: 08/11/93 DATE OF DISCHARGE 01/29/94

DISCHARGE SUMMARY

The physical examination at the time the baby was discharged, shows that the baby's right extremity at the level of the hips, knees and ankles shows that there is some tone as well as responses. There may be some tight dorsiflexion, but in general within normal limits. The left lower extremity on the contrary, the baby shows generalized weakness and there is still clonus and the deep tendon reflexes are absent and in general shows that the baby as a degree of paralysis on the left leg. The rest of the examination shows the skin normal and head normal, eyes, ears, eyes and throat normal. The chest with some bilateral rhonchi. Abdomen was normal. Genitalia normal. He has been circumcised and the rest within normal limits.

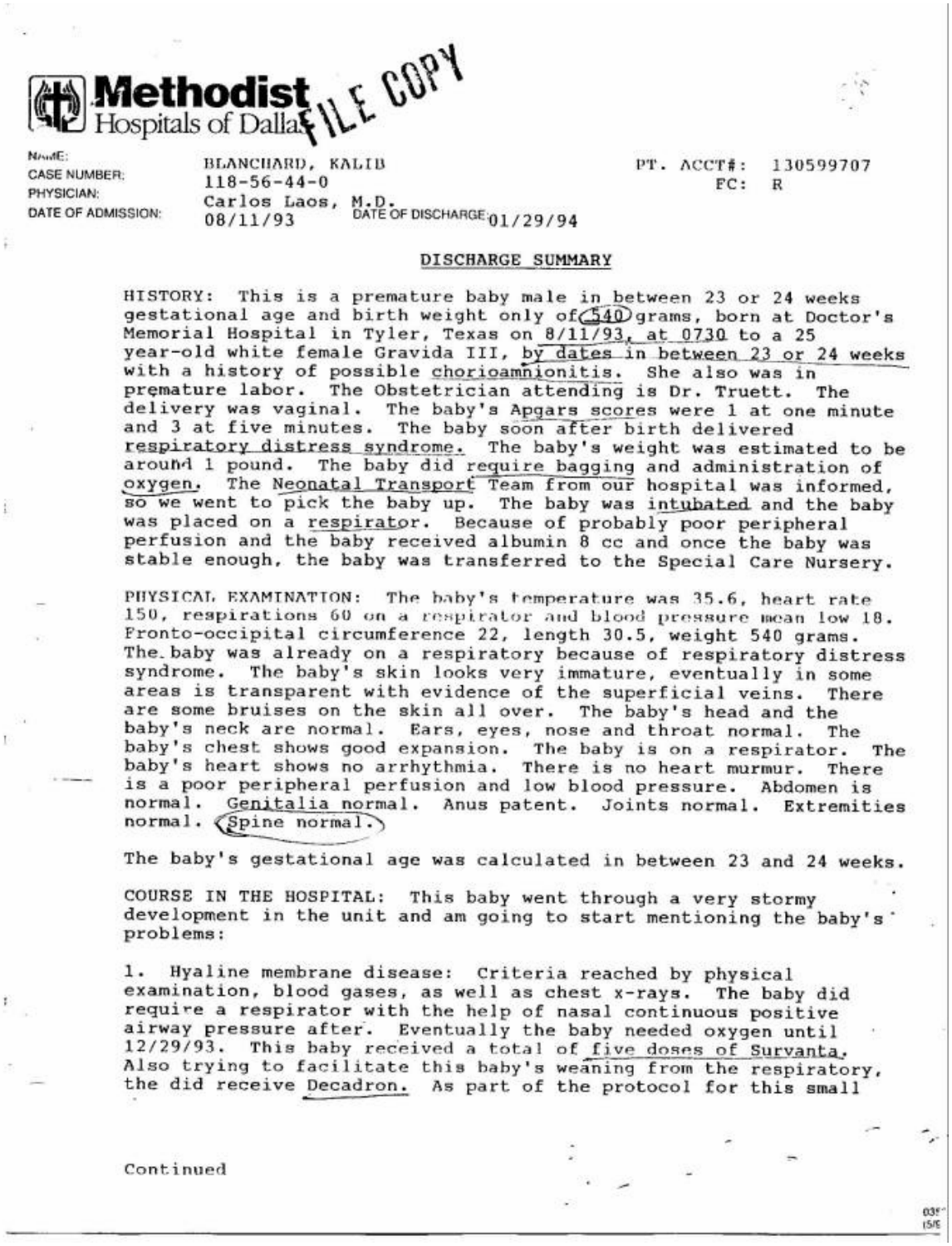
FINAL DIAGNOSES:

1. Premature baby male 23 to 24 weeks, 540 grams appropriate for gestational age.
2. Hyaline membrane disease.
3. Hypovolemic shock.
4. Thrombocytopenia.
5. Neonatal jaundice.
6. Retinopathy of prematurity.
7. Spinal cord injury at the level of T11, more on the left side than the right.
8. Bronchopulmonary dysplasia.
9. Apnea of prematurity.
10. Hypoproteinemia with hypoalbuminemia.
11. Hypernatremia.
12. Hyperkalemia.
13. Periventricular intraventricular hemorrhage.
14. Anemia of prematurity.
15. Septic shock.
16. Fungemia to Candida.
17. Patent ductus arteriosus.
18. Generalized skin breakdown with over-imposed fungal infection.
19. Hypokalemia.
20. Osteopenia with rickets of prematurity and hypophosphatemia.
21. Bilateral ventriculomegaly.
22. Phimosis with fistula.
23. Left lower quadrant ventral hernia.
24. Bilateral inguinal hernia.
25. Suspected gastroesophageal reflux.

CONSULTANTS: Dr. Chakmakjian - Pediatric Cardiologist.
Dr. Greene - Urologist.
DR. ROY ELTERMAN - Pediatric Neurologist

Continued

Figure 2a – first page of Discharge Summary



Methodist Hospitals of Dallas FILE COPY

NAME: BLANCHARD, KALIB PT. ACCT#: 130599707
CASE NUMBER: 118-56-44-0 FC: R
PHYSICIAN: Carlos Laos, M.D.
DATE OF ADMISSION: 08/11/93 DATE OF DISCHARGE: 01/29/94

DISCHARGE SUMMARY

HISTORY: This is a premature baby male in between 23 or 24 weeks gestational age and birth weight only of 540 grams, born at Doctor's Memorial Hospital in Tyler, Texas on 8/11/93, at 0730 to a 25 year-old white female Gravida III, by dates in between 23 or 24 weeks with a history of possible chorioamnionitis. She also was in premature labor. The Obstetrician attending is Dr. Truett. The delivery was vaginal. The baby's Apgars scores were 1 at one minute and 3 at five minutes. The baby soon after birth delivered respiratory distress syndrome. The baby's weight was estimated to be around 1 pound. The baby did require bagging and administration of oxygen. The Neonatal Transport Team from our hospital was informed, so we went to pick the baby up. The baby was intubated and the baby was placed on a respirator. Because of probably poor peripheral perfusion and the baby received albumin 8 cc and once the baby was stable enough, the baby was transferred to the Special Care Nursery.

PHYSICAL EXAMINATION: The baby's temperature was 35.6, heart rate 150, respirations 60 on a respirator and blood pressure mean low 18. Fronto-occipital circumference 22, length 30.5, weight 540 grams. The baby was already on a respiratory because of respiratory distress syndrome. The baby's skin looks very immature, eventually in some areas is transparent with evidence of the superficial veins. There are some bruises on the skin all over. The baby's head and the baby's neck are normal. Ears, eyes, nose and throat normal. The baby's chest shows good expansion. The baby is on a respirator. The baby's heart shows no arrhythmia. There is no heart murmur. There is a poor peripheral perfusion and low blood pressure. Abdomen is normal. Genitalia normal. Anus patent. Joints normal. Extremities normal. Spine normal.

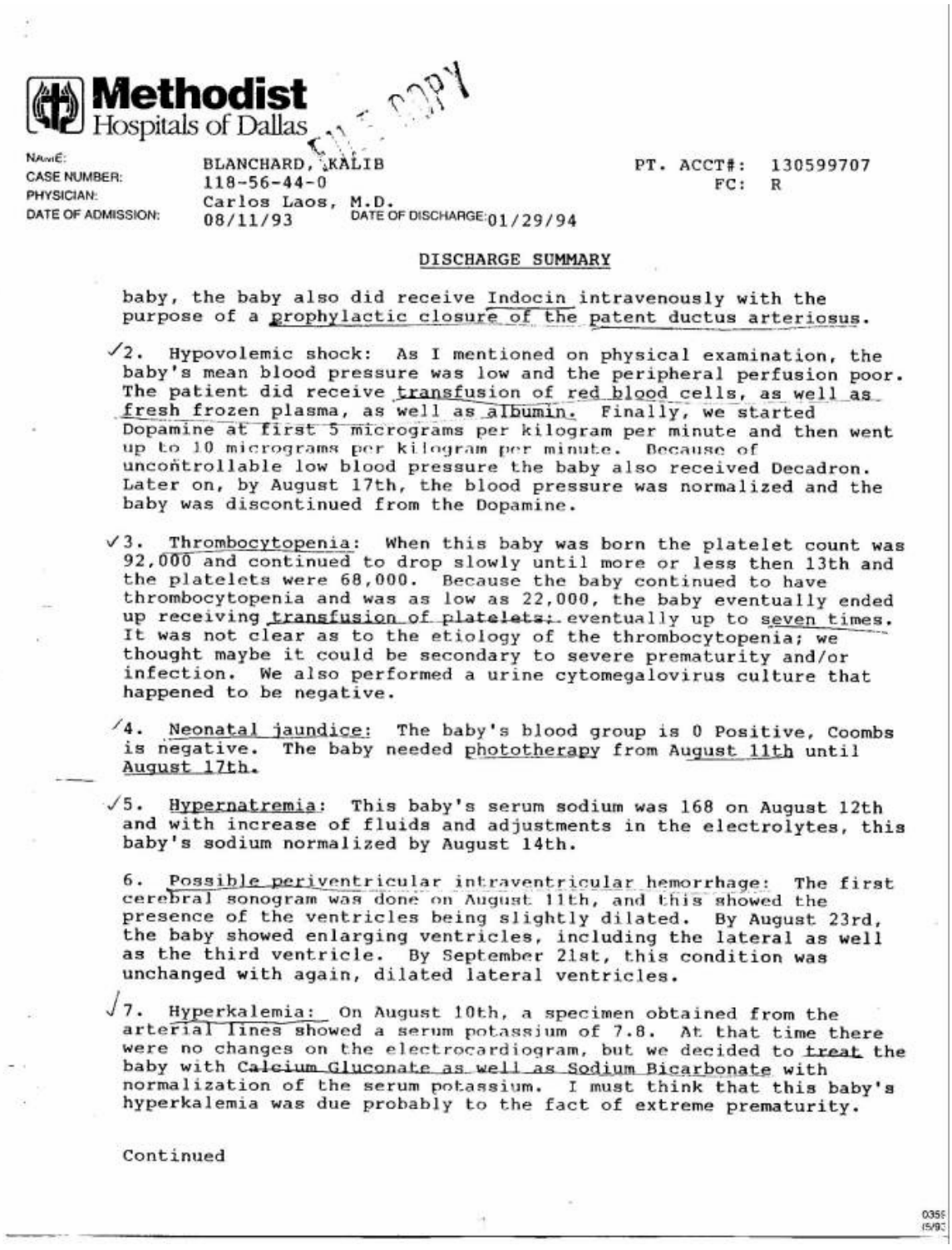
The baby's gestational age was calculated in between 23 and 24 weeks.

COURSE IN THE HOSPITAL: This baby went through a very stormy development in the unit and am going to start mentioning the baby's problems:

1. Hyaline membrane disease: Criteria reached by physical examination, blood gases, as well as chest x-rays. The baby did require a respirator with the help of nasal continuous positive airway pressure after. Eventually the baby needed oxygen until 12/29/93. This baby received a total of five doses of Survanta. Also trying to facilitate this baby's weaning from the respiratory, the did receive Decadron. As part of the protocol for this small

Continued

Figure 2b – Second page of Discharge Summary with Discharge Notes



Methodist Hospitals of Dallas

FILE COPY

NAME: BLANCHARD, KALIB PT. ACCT#: 130599707
CASE NUMBER: 118-56-44-0 FC: R
PHYSICIAN: Carlos Laos, M.D.
DATE OF ADMISSION: 08/11/93 DATE OF DISCHARGE: 01/29/94

DISCHARGE SUMMARY

baby, the baby also did receive Indocin intravenously with the purpose of a prophylactic closure of the patent ductus arteriosus.

2. Hypovolemic shock: As I mentioned on physical examination, the baby's mean blood pressure was low and the peripheral perfusion poor. The patient did receive transfusion of red blood cells, as well as fresh frozen plasma, as well as albumin. Finally, we started Dopamine at first 5 micrograms per kilogram per minute and then went up to 10 micrograms per kilogram per minute. Because of uncontrollable low blood pressure the baby also received Decadron. Later on, by August 17th, the blood pressure was normalized and the baby was discontinued from the Dopamine.

3. Thrombocytopenia: When this baby was born the platelet count was 92,000 and continued to drop slowly until more or less then 13th and the platelets were 68,000. Because the baby continued to have thrombocytopenia and was as low as 22,000, the baby eventually ended up receiving transfusion of platelets; eventually up to seven times. It was not clear as to the etiology of the thrombocytopenia; we thought maybe it could be secondary to severe prematurity and/or infection. We also performed a urine cytomegalovirus culture that happened to be negative.

4. Neonatal jaundice: The baby's blood group is O Positive, Coombs is negative. The baby needed phototherapy from August 11th until August 17th.

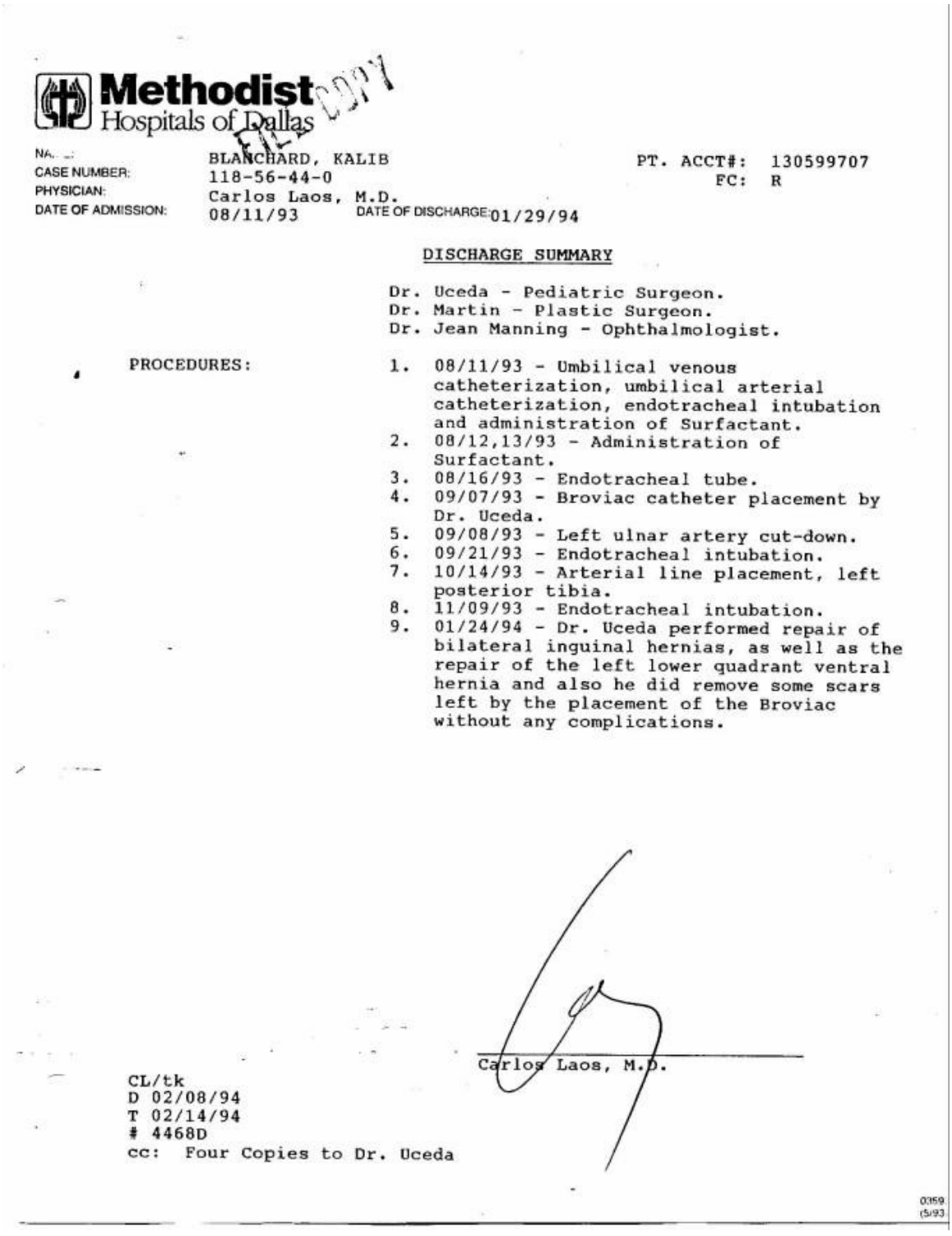
5. Hypernatremia: This baby's serum sodium was 168 on August 12th and with increase of fluids and adjustments in the electrolytes, this baby's sodium normalized by August 14th.

6. Possible periventricular intraventricular hemorrhage: The first cerebral sonogram was done on August 11th, and this showed the presence of the ventricles being slightly dilated. By August 23rd, the baby showed enlarging ventricles, including the lateral as well as the third ventricle. By September 21st, this condition was unchanged with again, dilated lateral ventricles.

7. Hyperkalemia: On August 10th, a specimen obtained from the arterial lines showed a serum potassium of 7.8. At that time there were no changes on the electrocardiogram, but we decided to treat the baby with Calcium Gluconate as well as Sodium Bicarbonate with normalization of the serum potassium. I must think that this baby's hyperkalemia was due probably to the fact of extreme prematurity.

Continued

Figure 2c – Third page of Discharge Summary; includes more Discharge Notes



Methodist Hospitals of Dallas

FILE COPY

NAME: BLANCHARD, KALIB PT. ACCT#: 130599707
CASE NUMBER: 118-56-44-0 FC: R
PHYSICIAN: Carlos Laos, M.D.
DATE OF ADMISSION: 08/11/93 DATE OF DISCHARGE: 01/29/94

DISCHARGE SUMMARY

Dr. Uceda - Pediatric Surgeon.
Dr. Martin - Plastic Surgeon.
Dr. Jean Manning - Ophthalmologist.

PROCEDURES:

1. 08/11/93 - Umbilical venous catheterization, umbilical arterial catheterization, endotracheal intubation and administration of Surfactant.
2. 08/12,13/93 - Administration of Surfactant.
3. 08/16/93 - Endotracheal tube.
4. 09/07/93 - Broviac catheter placement by Dr. Uceda.
5. 09/08/93 - Left ulnar artery cut-down.
6. 09/21/93 - Endotracheal intubation.
7. 10/14/93 - Arterial line placement, left posterior tibia.
8. 11/09/93 - Endotracheal intubation.
9. 01/24/94 - Dr. Uceda performed repair of bilateral inguinal hernias, as well as the repair of the left lower quadrant ventral hernia and also he did remove some scars left by the placement of the Broviac without any complications.

Carlos Laos, M.D.

CL/tk
D 02/08/94
T 02/14/94
4468D
cc: Four Copies to Dr. Uceda

0359
(5/93

Figure 2d – Last page of Discharge Summary; includes Procedure Notes

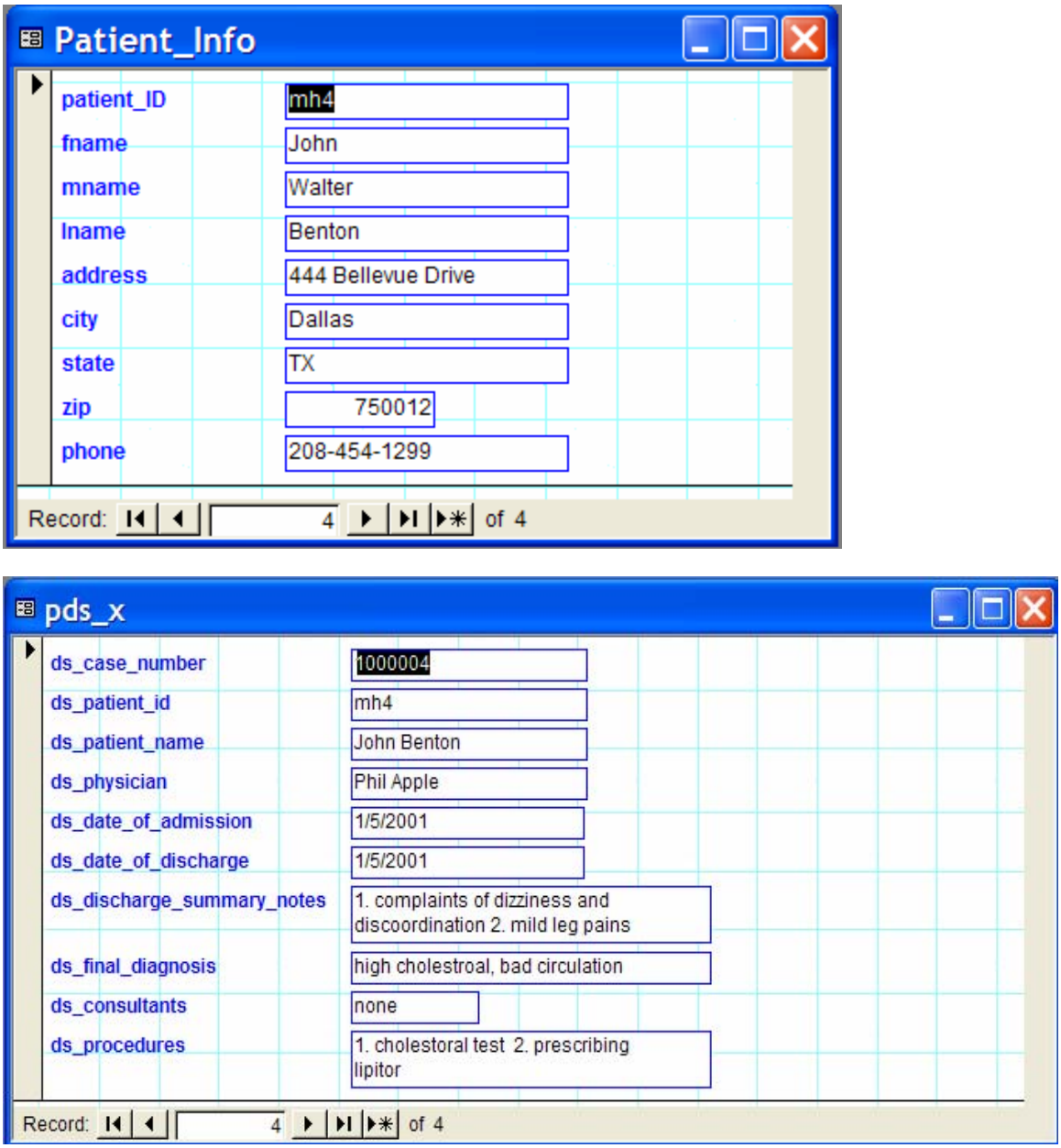


Figure 3: Screenshots of Access 2002 database. These forms were created to present information stored in the Patient_Info and Discharge_Summary tables, respectively. PDS_X is short for Patient Discharge Summary. Each label value is equal to the actual field heading in each table. These tables exist in the Methodist Hospital of Dallas database.

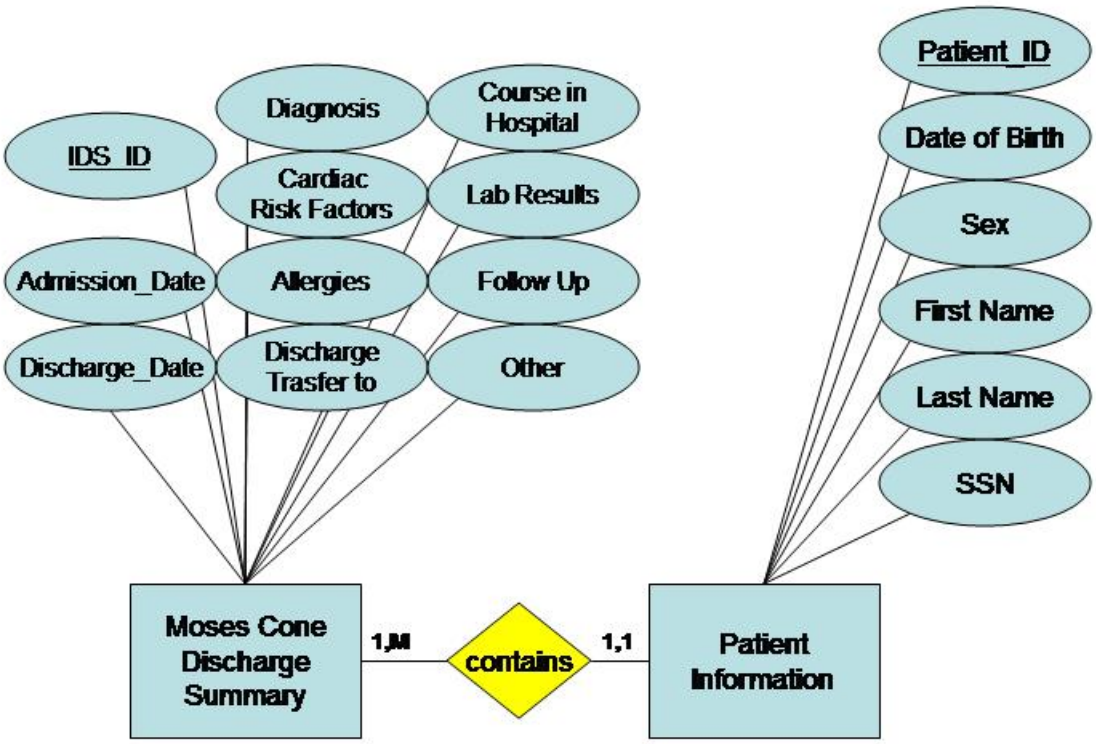


Figure 4: ER Diagram of Moses Cone Hospital Data Repository. This shows how discharge summary information and health information is stored in the Moses Cone database.

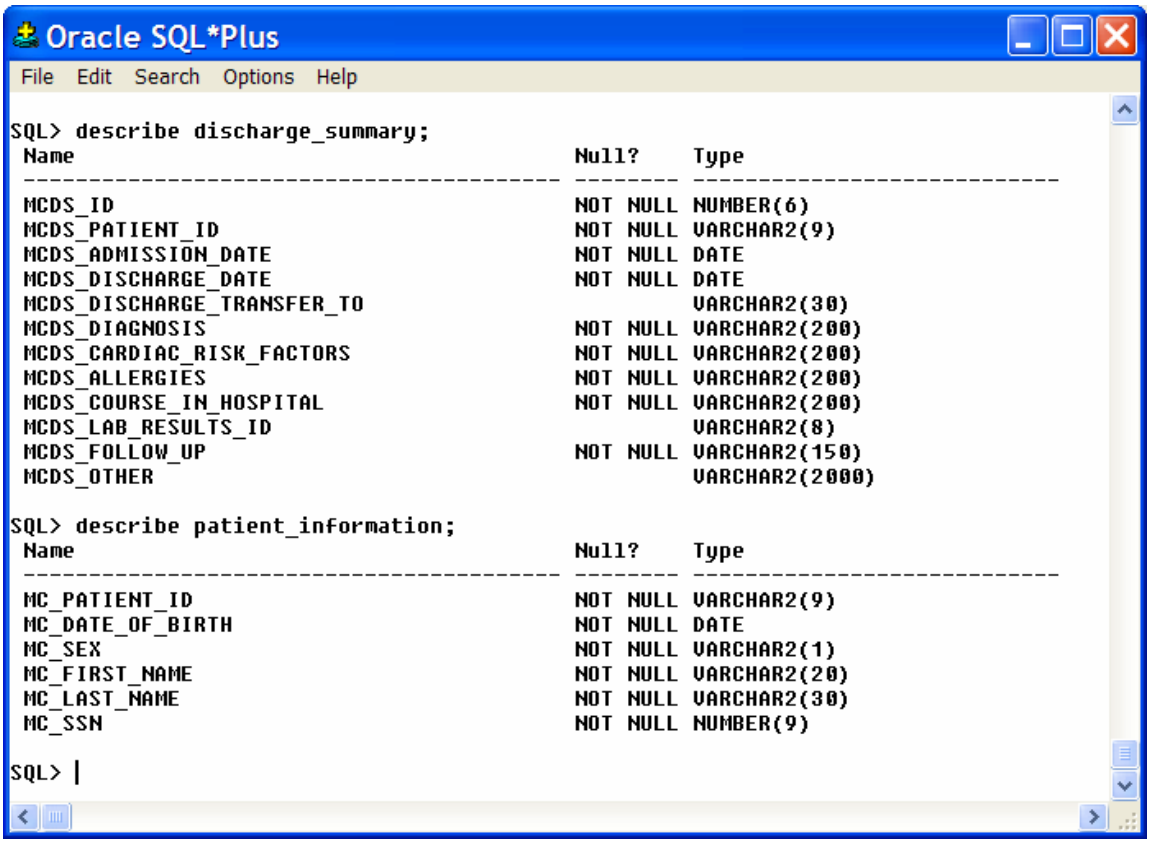


Figure 5a: Oracle Database Screenshot. Here is a screenshot of the Discharge Summary Table and the Patient Information table. Both exist in the Moses Cone Hospital Database.

Data Dictionary for DISCHARGE SUMMARY

Term:	Description:	Type / Format:
mcds_id	Unique ID for each discharge summary.	Number - 6 digits
mcds_patient_id	Unique ID for each patient. Foreign Key.	Text – 9 characters (mcp000001)
mcds_admission_date	Date that patient was admitted.	DATE
mcds_discharge_date	Date that patient was discharged.	DATE
mcds_discharge_transfer_to	If patient was transfered, this is the location.	Text - 30 max char.
mcds_diagnosis	Diagnosis information	Text -- 200 max char.
mcds_cardiac_risk_factors	Cardiac risk factors, if any exist.	Text – 200 max char.
mcds_allergies	Listing of known allergies.	Text – 200 max char.
mcds_course_in_hospital	List of procedures and courses of action taken by hospital	Text – 200 max char.
mcds_lab_results_ID	Unique ID for the lab results. Foreign Key.	Text/Num – 8 char.
mcds_follow_up	Follow Up instructions and/or information.	Text – 150 char.
mcds_other	If there is any further information, it is placed here	Text – 2000

Data Dictionary for PATIENT INFORMATION

Term:	Description:	Type / Format:
mc_patient_id	Unique ID for each patient	Text - 9 char. (mcp000001)
mc_date_of_birth	Date of birth of patient	DATE
mc_first_name	First name of patient	Text - 20 characters
mc_last_name	Last name of patient	Text - 30 characters
mc_ssn	Social security number of patient	Number - 10 digits

Figure 5b: Data Dictionary for Moses Cone Discharge Summary & Patient Information tables.

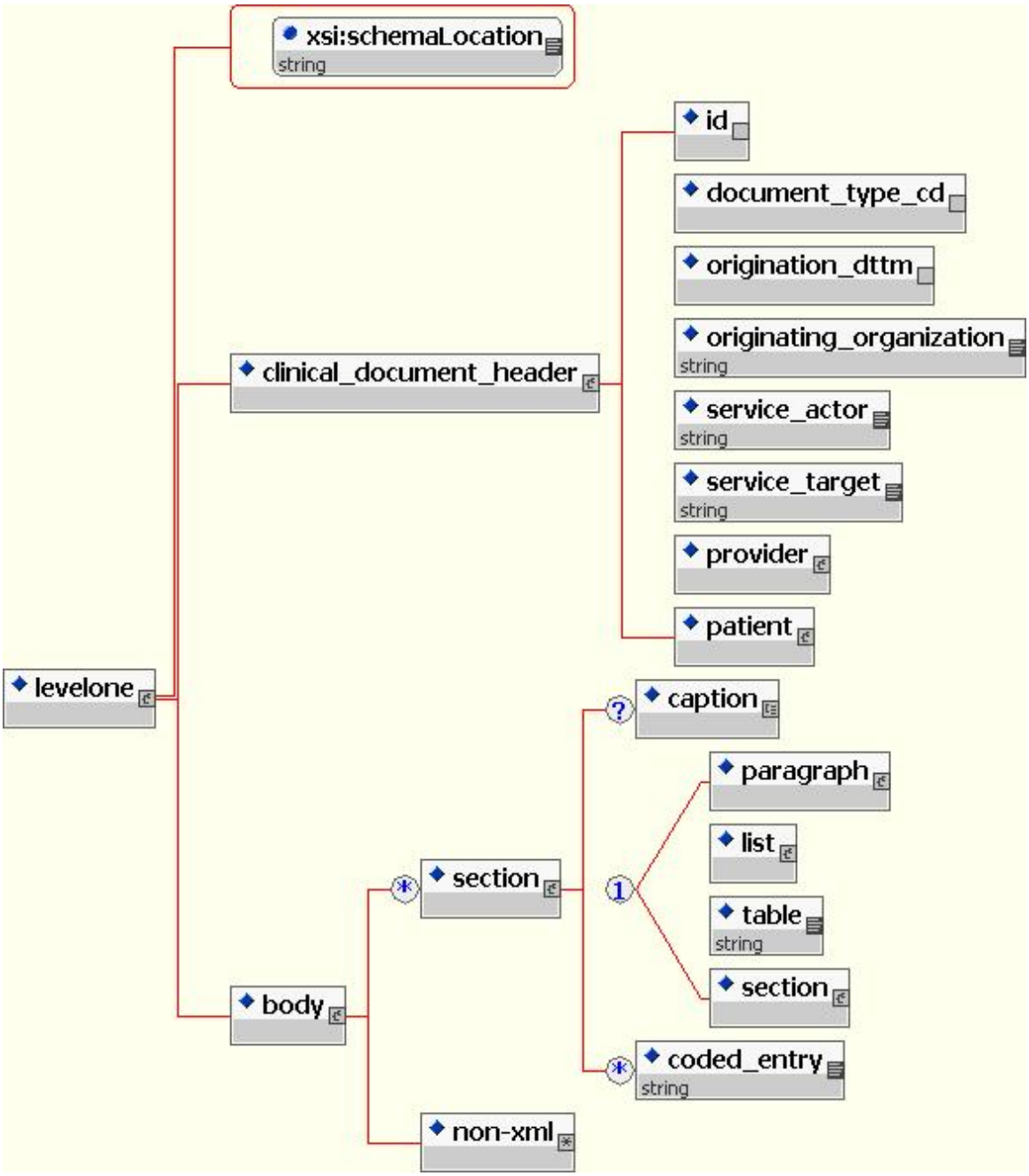


Figure 6: Graphic Model of Level One DTD. This is a graphic representation of the CDA Level One DTD. It includes all top-level elements, as well as some lower level elements. This is not representative of the entire schema.

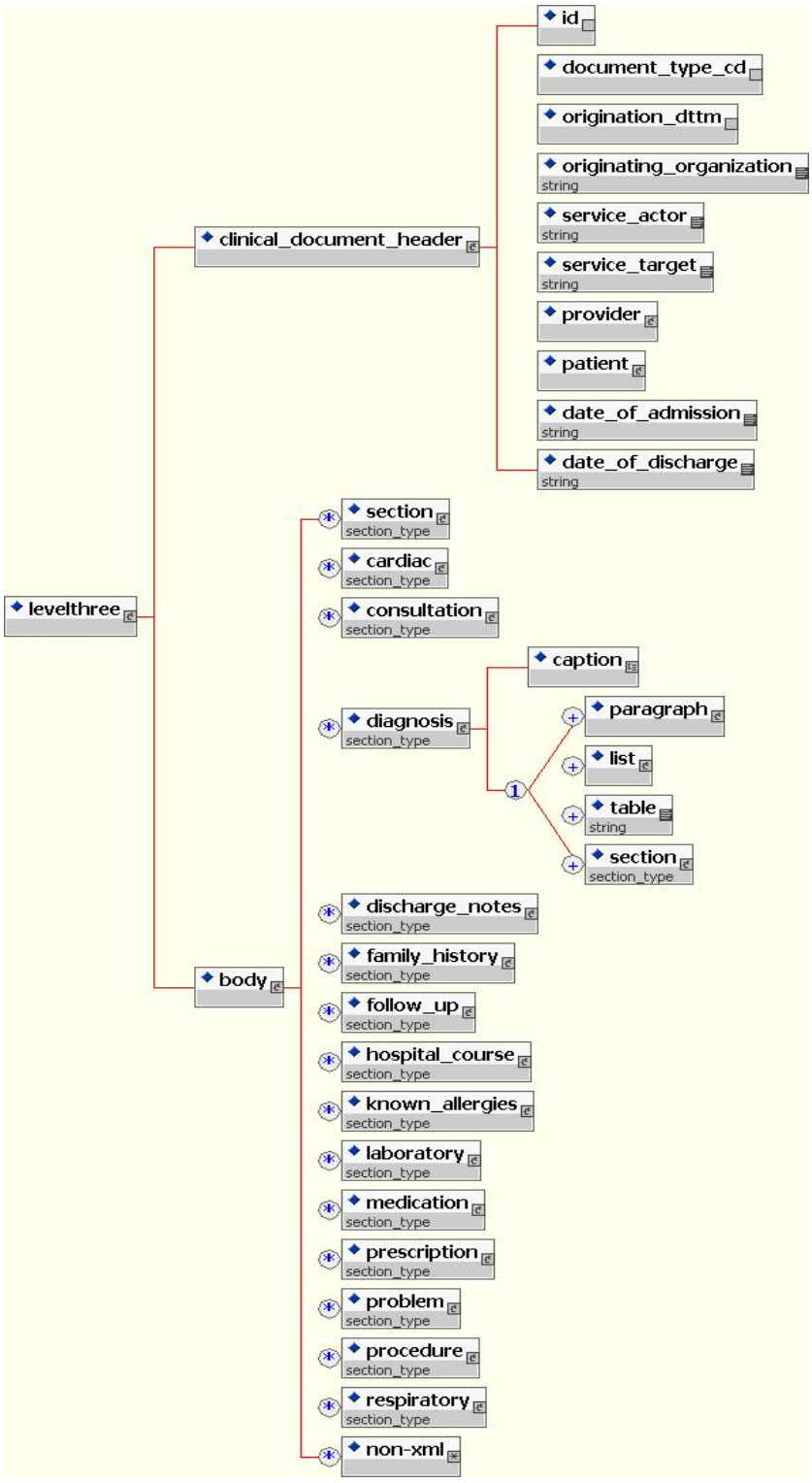


Figure 7: Graphic Model of Level Three schema. This is a graphic representation of the CDA Level Three schema. This schema was created to add machine-readable features to the XML structure. It builds on the Level One schema.

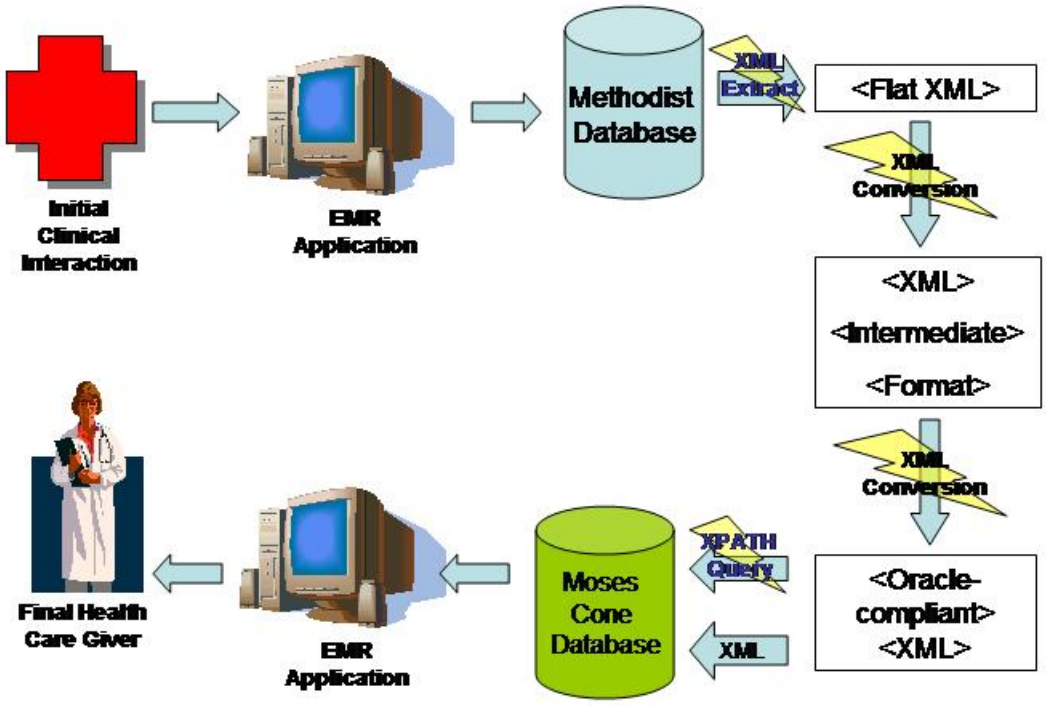


Figure 8: Data Flow Diagram. This diagram shows the path that the discharge summary information must take to go from Methodist Hospital of Dallas to Moses Cone Hospital.

```
SELECT ds_case_number, e.ch_document_type_cd, e.ch_origination_dttm,
e.ch_originating_organization, e.ch_service_actor, e.ch_service_target,
e.ch_provider_type, e.ch_provider_fname, e.ch_provider_mname,
e.ch_provider_lname, e.ch_patient_type, p.ds_patient_id, i.fname, i.mname,
i.lname, ds_physician, ds_date_of_admission, ds_date_of_discharge,
ds_discharge_summary_notes, ds_final_diagnosis, ds_consultants,
ds_procedures
FROM pds_x AS p, Patient_Info AS i, XML_export_clinical_header_data AS e
WHERE (((p.ds_case_number)=1000004) And ((p.ds_patient_id)=i.Patient_ID)
And ((e.ch_id)=p.ds_case_number));
```

Figure 9a: SQL statement to merge necessary tables in Methodist Hospital. This SQL statement was used in order to create one view of all of the data for the Discharge Summary XML document. The SQL was manipulated directly in order to create a specific summary of John Benton's discharge record.

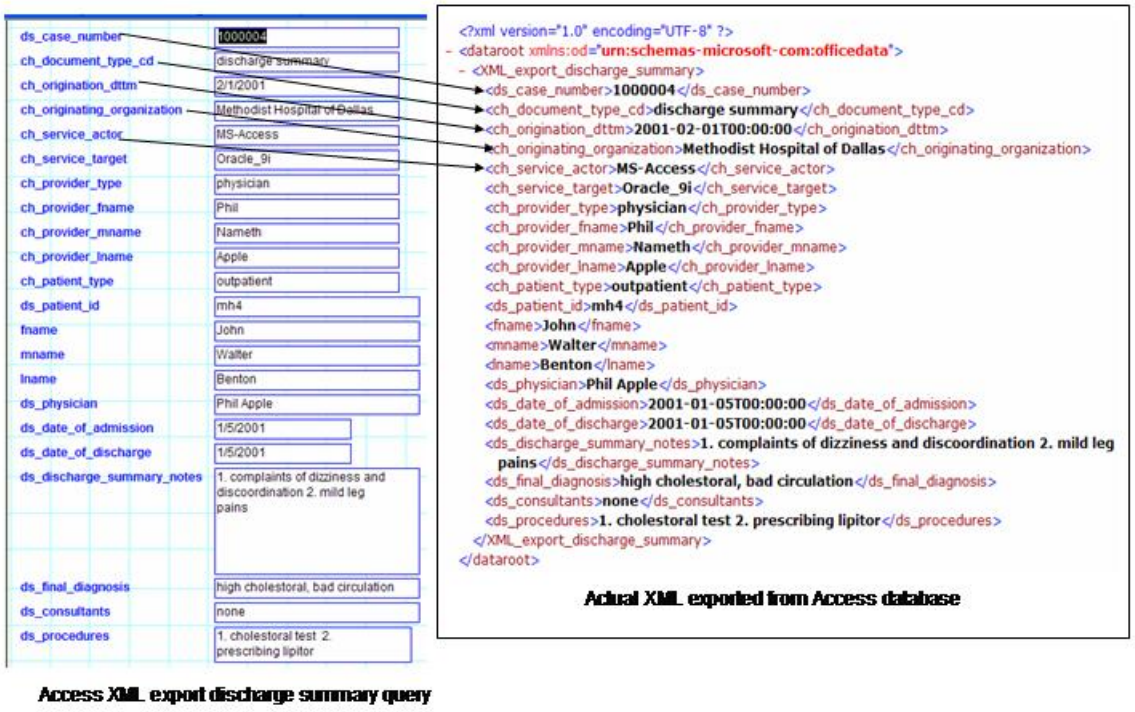


Figure 9b: XML export functionality of Access 2002. This shows how data is inserted into an XML document via the Access XML export Function.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <dataroot xmlns:od="urn:schemas-microsoft-com:officedata">
- <XML_export_discharge_summary>
<ds_case_number>1000004</ds_case_number>
<ch_document_type_cd>discharge summary</ch_document_type_cd>
<ch_origination_dttm>2001-02-01T00:00:00</ch_origination_dttm>
<ch_originating_organization>Methodist Hospital of
Dallas</ch_originating_organization>
<ch_service_actor>MS-Access</ch_service_actor>
<ch_service_target>Oracle_9i</ch_service_target>
<ch_provider_type>physician</ch_provider_type>
<ch_provider_fname>Phil</ch_provider_fname>
<ch_provider_mname>Nameth</ch_provider_mname>
<ch_provider_lname>Apple</ch_provider_lname>
<ch_patient_type>outpatient</ch_patient_type>
<ds_patient_id>mh4</ds_patient_id>
<fname>John</fname>
<mname>Walter</mname>
<lname>Benton</lname>
<ds_physician>Phil Apple</ds_physician>
<ds_date_of_admission>2001-01-05T00:00:00</ds_date_of_admission>
<ds_date_of_discharge>2001-01-05T00:00:00</ds_date_of_discharge>
<ds_discharge_summary_notes>1. complaints of dizziness and
discoordination 2. mild leg pains</ds_discharge_summary_notes>
<ds_final_diagnosis>high cholestoral, bad circulation</ds_final_diagnosis>
<ds_consultants>none</ds_consultants>
<ds_procedures>1. cholestoral test 2. prescribing lipitor</ds_procedures>
</XML_export_discharge_summary>
</dataroot>
```

Figure 10a – XML produced by Access 2002 database. As you will notice, the *dataroot* element is at the top level. All of the information was taken from the *XML Export Discharge Summary Query*, and even uses this query name as the second level element. All of the other elements are named after their field headings in the Access tables.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <xsl:transform
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  version="1.0">
- <xsl:template match="dataroot">
- <xsl:element name="levelthree">
<xsl:apply-templates
  select="XML_export_discharge_summary" />
</xsl:element>
</xsl:template>
- <xsl:template match="XML_export_discharge_summary">
- <xsl:element name="clinical_document_header">
- <xsl:element name="id">
- <xsl:attribute name="EX">
<xsl:value-of select="ds_case_number" />
</xsl:attribute>
</xsl:element>
- <xsl:element name="document_type_cd">
- <xsl:attribute name="V">
<xsl:value-of select="ch_document_type_cd" />
</xsl:attribute>
</xsl:element>
- <xsl:element name="origination_dttm">
- <xsl:attribute name="V">
<xsl:value-of select="ch_origination_dttm" />
</xsl:attribute>
</xsl:element>
- <xsl:element name="originating_organization">
<xsl:value-of select="ch_originating_organization" />
</xsl:element>
- <xsl:element name="service_actor">
<xsl:value-of select="ch_service_actor" />
</xsl:element>
- <xsl:element name="service_target">
<xsl:value-of select="ch_service_target" />
</xsl:element>
- <xsl:element name="provider">
- <xsl:attribute name="type">
<xsl:value-of select="ch_provider_type" />
</xsl:attribute>
- <xsl:element name="fname">
```

```
<xsl:value-of select="ch_provider_fname" />
</xsl:element>
- <xsl:element name="mname">
<xsl:value-of select="ch_provider_mname" />
</xsl:element>
- <xsl:element name="lname">
<xsl:value-of select="ch_provider_lname" />
</xsl:element>
</xsl:element>
- <xsl:element name="patient">
- <xsl:attribute name="type">
<xsl:value-of select="ch_patient_type" />
</xsl:attribute>
- <xsl:element name="fname">
<xsl:value-of select="fname" />
</xsl:element>
- <xsl:element name="mname">
<xsl:value-of select="mname" />
</xsl:element>
- <xsl:element name="lname">
<xsl:value-of select="lname" />
</xsl:element>
</xsl:element>
- <xsl:element name="date_of_admission">
<xsl:value-of select="ds_date_of_admission" />
</xsl:element>
- <xsl:element name="date_of_discharge">
<xsl:value-of select="ds_date_of_discharge" />
</xsl:element>
</xsl:element>
- <xsl:element name="body">
- <xsl:element name="consultation">
- <xsl:element name="paragraph">
- <xsl:element name="content">
<xsl:value-of select="ds_consultants" />
</xsl:element>
</xsl:element>
</xsl:element>
- <xsl:element name="diagnosis">
- <xsl:element name="paragraph">
- <xsl:element name="content">
```

```
<xsl:value-of select="ds_final_diagnosis" />
</xsl:element>
</xsl:element>
</xsl:element>
- <xsl:element name="discharge_notes">
- <xsl:element name="paragraph">
- <xsl:element name="content">
<xsl:value-of select="ds_discharge_summary_notes" />
</xsl:element>
</xsl:element>
</xsl:element>
- <xsl:element name="procedure">
- <xsl:element name="paragraph">
- <xsl:element name="content">
<xsl:value-of select="ds_procedures" />
</xsl:element>
</xsl:element>
</xsl:element>
</xsl:element>
</xsl:template>
</xsl:transform>
```

Figure 10b – XSLT stylesheet for transforming the Access query output XML. By using this stylesheet, it is possible to convert the XML output produced by Access 2002 to an XML document that conforms to the Standard Discharge Summary Schema.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <levelthree>
- <clinical_document_header>
<id EX="1000004" />
<document_type_cd V="discharge summary" />
<origination_dttm V="2001-02-01T00:00:00" />
<originating_organization>Methodist Hospital of
Dallas</originating_organization>
<service_actor>MS-Access</service_actor>
<service_target>Oracle_9i</service_target>
- <provider type="physician">
<fname>Phil</fname>
<mname>Nameth</mname>
<lname>Apple</lname>
</provider>
- <patient type="outpatient">
<fname>John</fname>
<mname>Walter</mname>
<lname>Benton</lname>
</patient>
<date_of_admission>2001-01-05T00:00:00</date_of_admission>
<date_of_discharge>2001-01-05T00:00:00</date_of_discharge>
</clinical_document_header>
- <body>
- <consultation>
- <paragraph>
<content>none</content>
</paragraph>
</consultation>
- <diagnosis>
- <paragraph>
<content>high cholestoral, bad circulation</content>
</paragraph>
</diagnosis>
- <discharge_notes>
- <paragraph>
<content>1. complaints of dizziness and discoordination 2. mild leg
pains</content>
</paragraph>
</discharge_notes>
- <procedure>
- <paragraph>
<content>1. cholestoral test 2. prescribing lipitor</content>
</paragraph>
</procedure>
</body>
</levelthree>
```

Figure 10c – XML output from XSLT transformation. This is the resulting XML document after the XSLT transformation. It conforms to the Standard Discharge Summary Schema.

Appendix B

Oracle Discharge Summary Fields:	Corresponding XML Schema elements:
MCDS_ID	*None*
Patient_ID	*None*
Admission_Date	Date_of_Admission
Discharge_Date	Date_of_Discharge
Discharge_Transfer_to	*None*
Diagnosis	Diagnosis
Cardiac_Risk_Factors	Cardiac
Allergies	Known Allergies
Course_in_Hospital	Procedure
Lab_Results_ID	*None*
Follow_Up	Follow_up
Other	*(any other fields that are left)*

Figure 1: Conceptual Map between Oracle Discharge Summary Table and the Standard Discharge Summary Schema.

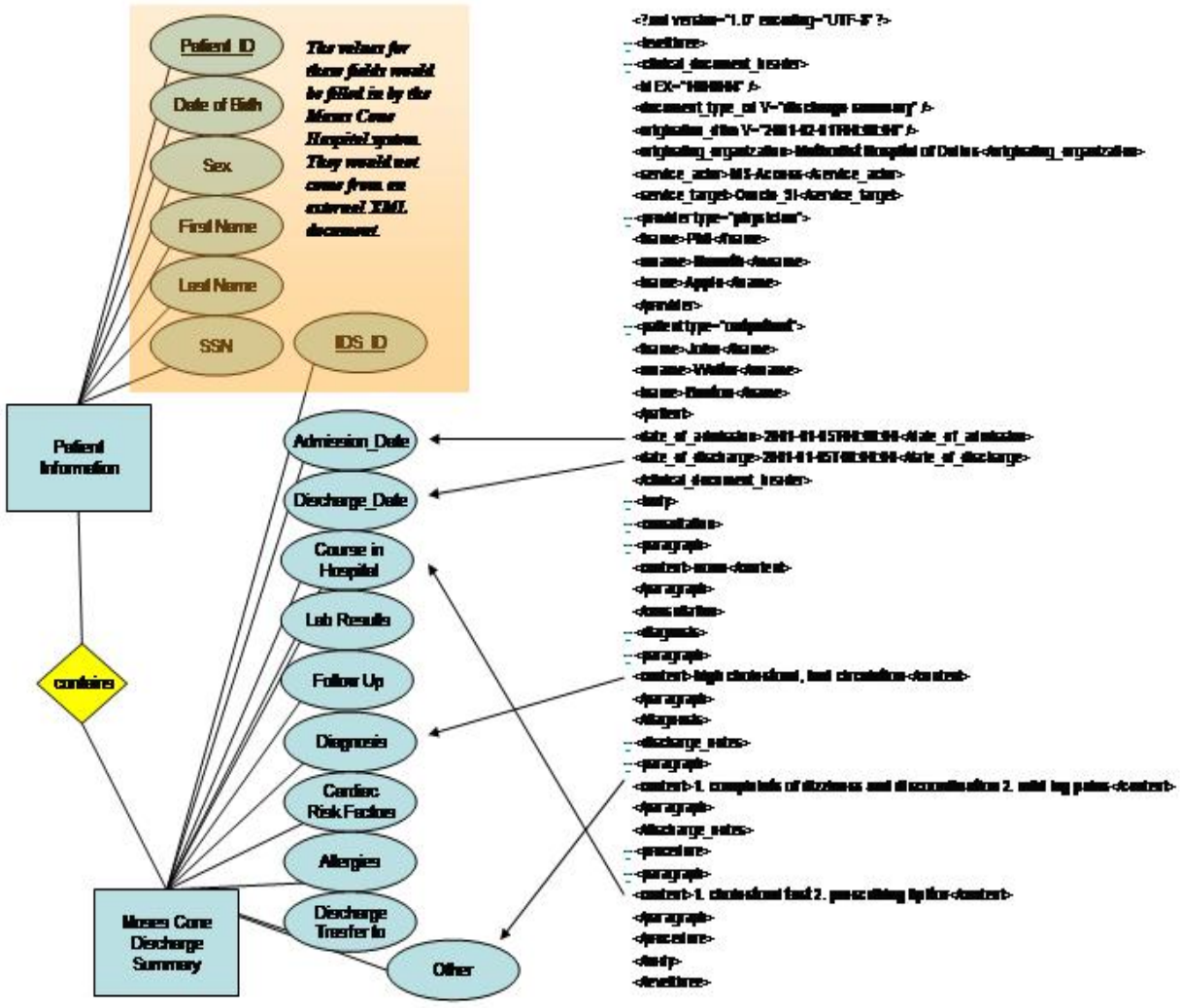


Figure 2: XML > Oracle RDBMS conceptual mapping. The solution follows this conceptual mapping in order to direct data into the appropriate fields. It is important to note that this is the Methodist XML document. Thus, some of the fields in the SDDS are not present.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <develthree>
  - <clinical_document_header>
      <id EX="1000004" />
      <document_type_cd V="discharge summary" />
      <origination_dttm V="2001-02-01T00:00:00" />
      <originating_organization>Methodist Hospital of Dallas</originating_organization>
```

Methodist Hospital XML document

Same XML document with necessary attributes

```
<?xml version="1.0" encoding="UTF-8" ?>
- <develthree xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="localhost:8080/home/MOSESCONE/xsd/discharge_summary.xsd">
  - <clinical_document_header>
      <id EX="1000004" />
      <document_type_cd V="discharge summary" />
      <origination_dttm V="2001-02-01T00:00:00" />
      <originating_organization>Methodist Hospital of Dallas</originating_organization>
```

Figure 3: Diagram of attribute addition executed by Oracle in order to import XML document into its domain. Above is a section of the SDSS-conformant XML document sent by Methodist Hospital. In order for Oracle to validate and save this XML document, the *xmlns:xsi* and *xsi:noNamespaceSchemaLocation* attributes must be added. They have been highlighted above.

```
INSERT INTO ds_tab Values (2, xmltype('<?xml version="1.0" encoding="WINDOWS-1252"?>
<levelthree xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="localhost:8080/home/MOSESCONE/xsd/discharge_summary.xsd">...</levelthree>')
 )
;
```

```
insert into discharge_summary_que VALUES ( '200002', 'mcp000011',
(SELECT extract(value(ds, '/levelthree/clinical_document_header/date_of_admission') FROM ds_tab where ID=2),
(SELECT extract(value(ds, '/levelthree/clinical_document_header/date_of_discharge') FROM ds_tab where ID=2),
null,
(SELECT extract(value(ds, '/levelthree/body/diagnosis/paragraph/content') FROM ds_tab where ID=2),
(SELECT extract(value(ds, '/levelthree/body/cardiac/paragraph/content') FROM ds_tab where ID=2),
(SELECT extract(value(ds, '/levelthree/body/known_allergies/paragraph/content') FROM ds_tab where ID=2),
(SELECT extract(value(ds, '/levelthree/body/procedure/paragraph/content') FROM ds_tab where ID=2),
(SELECT extract(value(ds, '/levelthree/body/laboratory/paragraph/content') FROM ds_tab where ID=2),
(SELECT extract(value(ds, '/levelthree/body/follow_up/paragraph/content') FROM ds_tab where ID=2),
(SELECT extract(value(ds, '/levelthree/body/discharge_notes/paragraph/content') FROM ds_tab where ID=2)
)
;
```

Figure 4a, 4b: Oracle Scripts. The above script is a sample of the SQL script code that is used to store an XML document natively in the Oracle repository. The '…' denotes where the XML document code would exist. The bottom script is a sample of the SQL / XPATH query used to insert XML data into the relational table.

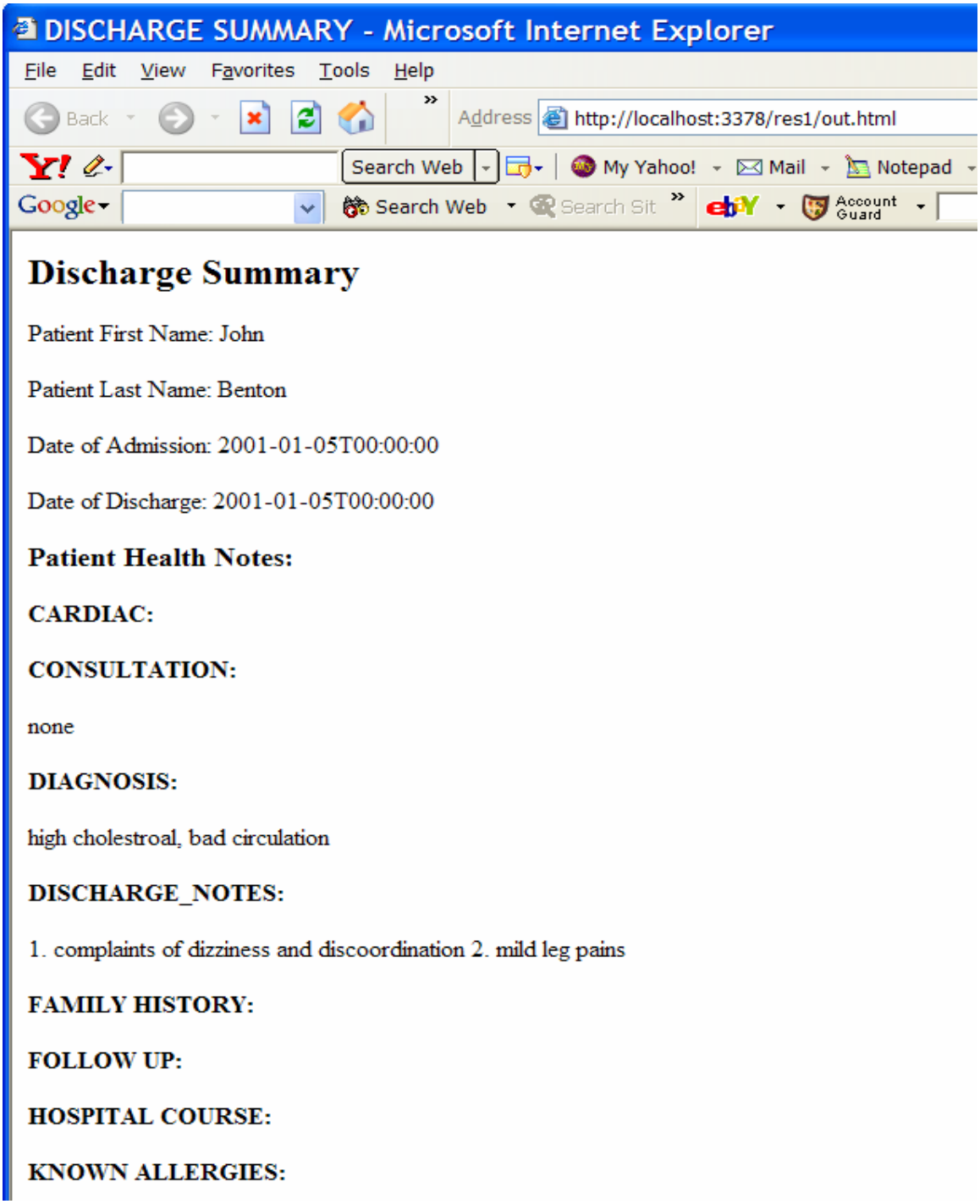


Figure 5: Presentation Layer. When XML health data is transmitted, it is important to include a presentation layer (usually in the form of an XSL Stylesheet) in order to format the presentation of the data in the case that it is not machine readable (or, the receiving target does not use Level Three machine-readability). This is an example of the XML discharge summary that has been styled as HTML by an XSL stylesheet.

References

Amer-Yahia, Sihem, and Srivastava, Divesh. "A Mapping Schema and Interface for XML Stores." *WIDM '02 Proceedings.* November 2002. p. 23-30.

Anderson, James G., Rainey, Michelle R., and Eysenbach, Gunther. "The Impact of CyberHealthcare on the Physician-Patient Relationship." *Journal of Medical Systems.* Vol. 27, No. 1, February 2003. p. 67-84.

Araki, Kenji, Ohashi, Katsuhiro, Yamazaki, Shunji, Hirose, Yasuyuki, Yamashita, Yoshinori, Yamamoto, Ryuichi, Minagawa, Kazushi, Sakamoto, Norihiro, and Yoshihara, Hiroyuki. "Medical Markup Language (MML) for XML-based Hospital Information Interchange." *Journal of Medical Systems.* Vol. 24, No. 3, 2000. p. 195-211.

Bates, David W., Ebell, Mark, Gotlieb, Edward, Zapp, John, and Mullins, H.C. "A Proposal for Electronic Medical Records in U.S. Primary Care." *Journal of the American Medical Informatics Association.* Volume 10, Number 1. Jan/Feb 2003. p. 1-10.

Chan, Wesley, Centiu, Cristian, and Morris, Jonathan. "Uniform Data Standards for Capturing Patient Medical Record Information at the Point of Care." Volume 13, Number 3, Fall 1999. p. 85-95.

Chapman, Sherwood. "XML and HIPAA: The Data Wave of the Future?" *Health Management Technology.* February 2003.

Clinical Document Architecture – PowerPoint Presentation. Available at http://www.hl7.org. 2003.

Denton, Ira C. "Will Patients Use Electronic Personal Health Records? Responses from a Real-Life Experience." *Journal of Healthcare Information Management.* Volume 15, Number 3, Fall 2001. p. 251-9.

Grimson, Jane, Grimson, William, and Hasselbring, Wilhelm. "The SI Challenge in Health Care." *Communications of the ACM.* Volume 43, No. 6. June 2000. p. 49-55.

Hassling, Linda, Babic, Ankica, Lonn, Urban, and Casimir-Ahn, Henrik. "A Web-Based Patient Information System—Identification of Patients' Information Needs." *Journal of Medical Systems.* Vol. 27, No.3, June 2003. p. 247-257.

HIMSS Standards Insight. "An Analysis of Health Information Standards and Development Initiatives." This document is available at http://www.hl7.org. July 2003.

HL7 Patient Record Architecture Draft. HL7 SGML/XML SIG. This document is available at http://www.hl7.org. August 3, 1999.

"Key Capabilities of an Electronic Health Record System: Letter Report." Institute of Medecine. This document is available at http://www.hl7.org. July 2003.

Kilbridge, Peter. "Crossing the Chasm with Information Technology: Bridging the Quality Gap in Health Care." *California HealthCare Foundation Report.* Produced by First Consulting Group. July 2002. p. 1-33.

Kulvatunyou, Boonserm, Ivezic, Nened, Martin, Monica, and Jones, Albert T. "A Business-to-Business Interoperability Testbed: An Overview." *ACM Conference Proceedings '00.* Jan/Feb 2000. p. 195-204.

Larsen, Ed. "The EHR Systems Functional Model and Standard Initiative." *HL7 News: The Official Publication of Health Level Seven, Inc.* August 2003.

Lee, K.P., and Jingkun, Hu. "XML Schema Representation of DICOM Structured Reporting." *Journal of the American Medical Informatics Association.* Volume 10, Number 2. Mar/Apr 2003. p. 213-23.

Loscio, Bernadette Farias, Salgado, Ana C., and Galvao, Luciano do Rego. "Conceptual Modeling of XML Schemas." *WIDM '03 Proceedings.* November 2003. p. 102-5.

Moen, William E. "Mapping the Interoperability Landscape for Networked Information Retrieval." *JDCL `01,* June 24-28, p. 50-51.

Nazi, Kim M. "The Journey to e-Health: VA Healthcare Network Upstate New York (VISN2)." *Journal of Medical Systems.* Vol. 27, No. 1, February 2003. p. 35-45.

Paepcke, Andreas, Chang, Chen-Chuan K., Garcia-Molina, Hector, and Winograd, Terry. "Interoperability for Digital Libraries Worldwide." *Communications of the ACM.* Volume 41, Number 4. April 1998. p. 33-43.

Ross, Stephen E., and Lin, Chen-Tan. "The Effects of Promoting Patient Access to Medical Records: A Review." *Journal of the American Medical Informatics Association.* Volume 10, Number 2. Mar/Apr 2003. p. 129-38.

Simone, Carla, Mark, Gloria, and Giubbilei, Dario. “Interoperability as a Means of Articulation Work.” *WACC ’99 Proceedings.* 1999. p. 39-48.

Strobel, Michael. “Communication Design for Electronic Negotiations on the Basis of XML Schema.” *WWW10 Proceedings.* May 2001. p. 9-20.

Su, Stanley and Lam, Herman. “An Extensible Knowledge Base Management System for Supporting Rule-based Interoperability.” *CIKM ’95 Proceedings.* 1995. p.1-10.

Walker, Rosemary, Dieter, Michael, Panko, Walter, and Valenta, Annette. “What It Will Take to Create New Internet Initiatives in Health Care.” *Journal of Medical Systems.* Vol. 27, No. 1, February 2003. p. 95-103.

Wileden, Jack C., Wolf, Alexander L., Rosenblatt, William R., and Tarr, Peri L. “Specification Level Interoperability.” *Communications of the ACM.* May 1991. Volume 34, No. 3. p. 72-88.